

PHP经典编程265例

快速服务：微博、QQ在线服务

自学视频：40集大型多媒体自学视频

海量资源：模块库、案例库、素材库、题库

潘凯华　李　慧　刘　欣　等编著

本书提供了内容丰富的配套资源，可以登录www.tup.com.cn，找到本书后，在该页面的“网络资源”超链接处下载。也可以访问本书的新浪微博，根据提示链接下载。

清华大学出版社

PHP经典编程265例

潘凯华　李　慧　刘　欣　等编著

（模块库、案例库、素材库、题库）

（微博、QQ、论坛技术支持）

清华大学出版社

北　京

内 容 简 介

本书以基础知识为框架，介绍了各部分知识所对应的常用开发实例，并进行透彻地解析。内容包括 PHP 基础、函数、PHP 流程控制语句、Web 技术、MySQL 数据库、PHP 数据库编程、字符串高级处理、PHP 数组应用、日期和时间的处理、图形图像处理、文件目录处理、面向对象编程、PDO 数据库抽象层、Smarty 模板、ThinkPHP 框架和 PHP 的字符编码。

本书所精选的实例都是一线开发人员在实际项目中的积累，对这些实例进行了技术上的解析，并给出了详细的实现过程。读者通过对本书的学习，能够提高开发能力，积累开发经验。

本书提供了大量的源程序、素材，提供了相关的模块库、案例库、素材库、题库等多种形式辅助学习资料，还提供迅速及时的微博、QQ、论坛等技术支持。

本书内容详尽，实例丰富，非常适合作为零基础学习人员的学习用书和大中专院校师生的学习教材，也适合作为相关培训机构的师生和软件开发人员的参考资料。

图书在版编目（CIP）数据

PHP 经典编程 265 例/潘凯华，李慧，刘欣等编著. —北京：清华大学出版社，2012.2

ISBN 978-7-302-27315-8

I. ①P…　II. ①潘…　②李…　③刘…　III. ①PHP 语言-程序设计　IV. ①TP312

中国版本图书馆 CIP 数据核字（2011）第 236934 号

责任编辑：赵洛育　刘利民
版式设计：文森时代
责任校对：姜　彦
责任印制：
出版发行：清华大学出版社

网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185mm×260mm　**印　　张**：28.25　**字　　数**：653 千字
版　　次：2012 年 2 月第 1 版　**印　　次**：2012 年 2 月第 1 次印刷
印　　数：1～5000
定　　价：49.80 元

产品编号：043925-01

前　言

Preface

学会站在巨人的肩膀上！

Web 项目开发的终极目标是完成满足用户需求的项目。一个项目往往包含复杂的功能。作为一名程序员，需要在有限的时间内实现它们，这对于一名新手显然并不容易。为何有开发经验的程序员编程效率会非常高？答案就是他们做过类似的程序，通过适当修改以前的代码就可以满足现在的要求。因此，如何快速积累编程经验就成了新手的当务之急。显然，单单依靠项目来积累速度是非常慢的。

本书图文并茂、难易并举，汇集了 265 个日常开发中经常使用的实例，内容涵盖 PHP 语言 Web 项目开发的方方面面。每个实例分成实例说明、实现过程和技术要点 3 部分进行讲解。通过学习本书，不但能快速掌握相关知识点，还能逐步提高编程能力。

本书内容

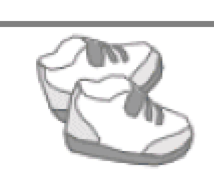
本书以基础知识结构为框架，每一部分知识都有范例举证，不但是对基础知识的巩固，更能够体现基础知识在实战开发中的应用。全书分为 16 章，内容包括 PHP 基础、函数、PHP 流程控制语句、Web 技术、MySQL 数据库、PHP 数据库编程、字符串高级处理、PHP 数组应用、日期和时间的处理、图形图像处理、文件目录处理、面向对象编程、PDO 数据库抽象层、Smarty 模板、ThinkPHP 框架和 PHP 的字符编码。

为了更清晰地阐述问题和给出问题的解决方案，本书设置了以下栏目。

☑ 实例说明：详细描述本实例的用途，并给出实例的运行效果图。
☑ 实现过程：逐步讲解如何解决本实例的问题，并给出关键代码、注意事项等。
☑ 技术要点：对本实例使用的关键技术进行总结，方便日后使用。

本书特色

☑ 贴近应用。本书精选的实例都是真正来自开发一线。以实例的形式进行讲解，更容易被读者接受。
☑ 横向链接。本书知识框架与《PHP 开发入门及项目实战》一书相对应，可以在使用《PHP 开发入门及项目实战》一书进行基础学习之后，再用本书丰富并提高技能。
☑ 解析透彻。本书对每个问题的相关知识进行细致讲解，并进行知识拓展，使读者不仅知其然而且知其所以然。
☑ 授人以渔。本书在讲解技术的同时还注重对读者能力的培养，使读者掌握分析问题与解决问题的能力。





Note

本书配套资源

本书提供了内容丰富的配套资源，包括源程序、素材，以及模块库、案例库、题库、素材库等多项辅助内容，读者朋友可以通过如下方式获取。

第 1 种方式：

（1）登录 www.tup.com.cn，在网页右上角的搜索文本框中输入本书书名（注意区分大小写和留出空格），或者输入本书关键字，或者输入本书 ISBN 号（注意去掉 ISBN 号间隔线“-”），单击“搜索”按钮。

（2）找到本书后单击超链接，在该书的网页下侧单击“网络资源”超链接，即可下载。

第 2 种方式：

访问本书的新浪微博 PHPbook，找到配套资源的链接地址进行下载。

读者人群

本书非常适合以下人员阅读：

☑ 从事 PHP 语言编程行业的开发人员
☑ 有一定语言基础，想进一步提高技能的人员
☑ 大中专院校的老师和学生
☑ 即将走上工作岗位的大学毕业生
☑ 相关培训机构的老师和学员
☑ PHP 语言编程爱好者

读者服务&本书勘误

读者在使用本书过程中遇到的所有问题，均可通过以下方式联系我们。

1．新浪微博：PHPbook。

及时发布读者答疑、本书勘误、配套资料更新等内容。

2．腾讯 QQ：4006751066。

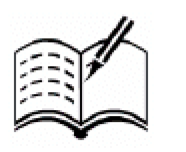
3．登录网站：www.mingribook.com，在论坛、勘误发布、读者纠错、技术支持、读者之家等栏目中的相关模块中提问、留言或查看。

本书作者

本书由明日科技组织编写，参加编写的有潘凯华、李慧、刘欣、陈丹丹、王国辉、张振坤、李伟、沈博、孙秀梅、曹飞飞、王雪、朱晓、赵永发、李鑫、高春艳、王小科、赵会东、李继业、赛奎春、杨丽、李丽、刘龄龄、王明招、孙茜、陈英、肖鑫和刘冠男等。由于作者水平有限，疏漏和不足之处在所难免，敬请广大读者朋友批评指正。

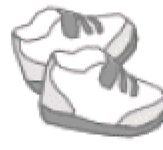
编　者

目　录



Contents

第 1 章　PHP 基础 1
实例 001　获取当前执行文件的名称........... 2
实例 002　计算一个圆形的面积 3
实例 003　当数字遇到字符串 4
实例 004　通过转义字符输出特殊字符串 ... 5
实例 005　通过 PHP5 新型字符串动态
输出 JavaScript 代码................... 6
实例 006　比较两个时间戳的大小.............. 7
实例 007　判断数字的奇偶性 8
实例 008　通过逻辑运算符判断用户
的权限 9
实例 009　获取 2000～2020 年中所有
的闰年 10
实例 010　自定义数字加密算法 11
实例 011　随机输出字符串....................... 12
实例 012　在页面中打印服务器时间.......... 13
实例 013　区分单引号和双引号 15
实例 014　前置运算符和后置运算符
的区别 15
实例 015　使用可变变量输出
“I Like PHP!” 16
第 2 章　函数.. 18
实例 016　自定义函数过滤字符串............ 19
实例 017　获取上传文件的后缀 20
实例 018　统一上传文件名称的大小写..... 21
实例 019　论坛内容的简单输出 22
实例 020　日期、时间的名称化输出......... 23
实例 021　倒计时 24
实例 022　中文图像验证码 26
实例 023　上传文件到服务器 27
实例 024　读取整个文本文件中的内容..... 29
实例 025　连接 MySQL 数据库服务器..... 31
实例 026　选择指定的 MySQL 数据库..... 32
实例 027　执行 SQL 语句......................... 34
实例 028　文本文件分页读取 35
实例 029　查询关键字描红 38
实例 030　分页显示图书信息 39
实例 031　一般用户注册 41
实例 032　带检测用户名的用户注册 42
实例 033　分步用户注册 45
第 3 章　PHP 流程控制语句 47
实例 034　员工生日提醒 48
实例 035　SWITCH 网页框架 49
实例 036　员工信息的批量删除 51
实例 037　do…while 语句循环读取
数据库中数据 52
实例 038　生成随机验证码 53
实例 039　图形计数器 55
实例 040　包含数据库连接文件 57
实例 041　健康生活提醒 58
实例 042　if 语句判断美女征婚条件 60
实例 043　网页版九九乘法表 62
实例 044　读取购物车中的数据 64
实例 045　多图片上传 65
实例 046　控制页面中表情图的输出 67
实例 047　控制页面中数据的输出数量 68
实例 048　考试成绩评定标准 69
第 4 章　Web 技术.................................... 71
实例 049　通过客户端 IP 限制投票次数... 72
实例 050　设计论坛登录界面 73
实例 051　可以上传图片的表单 74
实例 052　以文本域的形式显示数据信息... 76
实例 053　验证用户注册信息是否合理 77
实例 054　省市级联动菜单 79
实例 055　省市县级联动菜单 80
实例 056　实现复选框中的全选、
反选和不选 82
实例 057　上传图片预览 84

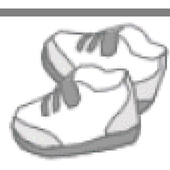
实例 058　通过下拉列表选择头像............86
实例 059　日期选择器..............................87
实例 060　在页面右下角弹出渐显的广告窗口..............................90
实例 061　树形导航菜单..........................91
实例 062　收缩式导航菜单......................93
实例 063　控制登录用户的过期时间.........95
实例 064　单点登录..................................97
实例 065　统计用户在线时间...................99
实例 066　限制用户访问网站的时间.......101
实例 067　SESSION 更换聊天室界面.....102
实例 068　掌控登录用户的权限.............103
实例 069　屏蔽页面刷新对计数器的影响..................................105
实例 070　SESSION 购物车...................106
实例 071　清理 SESSION 缓存提高网站访问的效率.....................109
实例 072　限制上传文件的大小..............111
实例 073　限制上传文件的类型.............113
实例 074　通过链接方式下载.................114
实例 075　上传多个文件到服务器..........115
实例 076　通过 header()函数进行下载....117
实例 077　重新定义上传文件的名称.......120

第 5 章　MySQL 数据库.......................121

实例 078　启动 MySQL 服务.................122
实例 079　连接 MySQL 服务器..............122
实例 080　关闭 MySQL 服务.................123
实例 081　创建 PHP 图书数据库............124
实例 082　选择 PHP 图书数据库............125
实例 083　删除 PHP 图书数据库............126
实例 084　在 PHP 图书数据库中创建图书信息表.......................127
实例 085　查看图书信息表.....................129
实例 086　修改图书信息表.....................130
实例 087　重命名图书信息表.................131
实例 088　删除图书信息表.....................132
实例 089　向图书信息表中添加数据.......132
实例 090　修改图书信息表中的数据.......133
实例 091　删除图书信息表中所有数据...135
实例 092　删除图书信息表中指定数据...136
实例 093　通过 phpMyAdmin 修改MySQL 用户密码..................137
实例 094　通过 phpMyAdmin 设置数据库、数据表编码.............139
实例 095　phpMyAdmin 操作数据库.......140
实例 096　phpMyAdmin 操作数据表.......141
实例 097　phpMyAdmin 操作数据..........143

第 6 章　PHP 数据库编程......................146

实例 098　通过 MySQL 函数访问数据库....147
实例 099　将数据以二进制形式上传到数据库.........................149
实例 100　查询日期型数据.....................152
实例 101　查询字符串............................154
实例 102　使用 MySQL 存储过程实现用户注册.........................155
实例 103　使用 MySQL 事务处理实现银行安全转账...................158
实例 104　避免输出中文字符串时出现乱码..................................161
实例 105　查询指定时间段的数据..........162
实例 106　查询从指定位置的前 N 条记录..................................164
实例 107　通过 PHP 面向过程实现数据分页..................................165
实例 108　通过 PHP 面向对象实现数据分页..................................170
实例 109　查询结果不显示重复记录......173
实例 110　Delete 语句删除图书信息.......174
实例 111　对统计结果进行排序..............175
实例 112　使用 select 子句进行多表查询...177
实例 113　合并多个结果集.....................178
实例 114　简单的嵌套查询.....................179
实例 115　用 in 查询表中的记录信息.....180
实例 116　使用聚集函数 sum()对学生成绩进行汇总.........................182
实例 117　使用聚集函数 avg 求学生的平均成绩.............................183
实例 118　使用聚集函数 min()求利润最少的商品.........................184
实例 119　使用聚集函数 max()求销售利润最高的商品.....................186
实例 120　使用聚集函数 count()求利润大于某值的数据.................188
实例 121　多表联合查询........................189



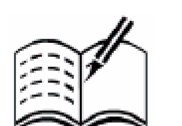
实例 122　left outer join 查询 190
实例 123　right outer join 查询 191
实例 124　利用 transform 分析数据 193
实例 125　使用格式化函数转换查询条件的数据类型 194
实例 126　在查询中使用日期函数 196
实例 127　一般搜索 198
实例 128　高级搜索 200
实例 129　程序员搜索引擎 202

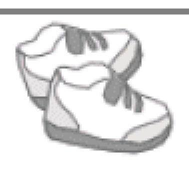
第 7 章　字符串高级处理 205
实例 130　统计帖子标题的长度 206
实例 131　购物车中数据的读取 206
实例 132　查询关键字描红 208
实例 133　统计查询关键字的出现次数 209
实例 134　去除帖子标题的首尾空格 210
实例 135　验证电话号码的格式是否正确 211
实例 136　验证 E-mail 地址格式是否正确 212
实例 137　超长文本的分页输出 213
实例 138　统一上传文件名称的大小写 215
实例 139　货币数据的格式化输出 216
实例 140　统一英文注册用户首字母大小写 217
实例 141　解决用 substr()函数对中文字符串截取时乱码 218
实例 142　将元素中指定位置的索引替换 219
实例 143　统计数组元素的个数 220
实例 144　去除数组中的重复元素 221
实例 145　判断字符串中是否存在指定子串 222
实例 146　合并数组 223
实例 147　拆分数组 224
实例 148　将时间和日期转换为时间戳 225
实例 149　网页闹钟 226

第 8 章　PHP 数组应用 228
实例 150　查看最便宜的图书 229
实例 151　随机抽取图书 229
实例 152　车牌摇号 230
实例 153　获取上传文件的数据 231
实例 154　$_FILES[]全局数组在文件上传中的应用 232
实例 155　图书信息逆向输出 233
实例 156　统计图书的数量 234
实例 157　获取图书馆中最受欢迎的 3 本图书 235
实例 158　生成在线考试题 236
实例 159　向购物车中添加商品 238
实例 160　查看购物车 240
实例 161　从购物车中移去指定商品 242
实例 162　修改商品购买数量 243
实例 163　清空购物车 244
实例 164　收银台结账 245

第 9 章　日期和时间的处理 248
实例 165　获取不同地区的当前时间 249
实例 166　计算考试时间 250
实例 167　输出中文格式的日期和时间 251
实例 168　检验日期和时间的有效性 252
实例 169　倒计时 253
实例 170　判断时间的早晚 253
实例 171　网页闹钟 254
实例 172　计算程序的运行时间 255

第 10 章　图形图像处理 257
实例 173　GD2 函数填充几何图形 258
实例 174　GD2 函数在照片上添加文字 259
实例 175　GD2 函数为图片添加文字水印 261
实例 176　GD2 函数为图片添加图像水印 263
实例 177　GD2 函数生成图形验证码 264
实例 178　折线图分析 2010 年的销售额 266
实例 179　柱形图分析编程词典销售比例 268
实例 180　饼形图分析 2010 年图书销量 269
实例 181　GD2 函数折线图分析网站月访问量走势 271
实例 182　GD2 函数柱状图分析编程词典满意度调查 273
实例 183　GD2 函数饼形图分析图书市场的份额 276





实例 184　柱状图展示编程词典 6、7 月份销售量 278
实例 185　柱状图展示编程词典上半年销量 280
实例 186　折线图分析 2010 年牛肉市场价格走势 282
实例 187　多饼形图区块分析 2010 年图书销量 283
实例 188　缩略图艺术库 285
实例 189　通过图像显示投票统计结果 288
实例 190　通过图像显示密码安全强度 290
实例 191　任意调整上传图片的大小 292

第 11 章　文件目录处理 295

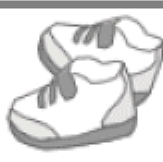
实例 192　文件操作汇总 296
实例 193　文件类型检测 298
实例 194　删除指定目录下的所有 ini 文件 301
实例 195　重新定义目录的名称 302
实例 196　获取磁盘分区的大小 304
实例 197　遍历指定目录下的所有文件 306
实例 198　可以屏蔽刷新功能的文本计数器 308
实例 199　从文本文件读取注册服务条款 310
实例 200　遍历、删除指定文件目录下的所有文件 311
实例 201　文件属性分析 314
实例 202　将文本文件中的内容存储到数据库中 315
实例 203　判断文件是否被改动 318
实例 204　目录操作汇总 319

第 12 章　面向对象编程 322

实例 205　使用类的属性保存数据库连接参数 323
实例 206　数据库连接类中定义数据库连接方法 325
实例 207　数据统计类中定义求数值平方的方法 326
实例 208　使用$this 关键字调用汽车类自身的方法 328
实例 209　学生类中使用构造方法为学生信息初始化 330
实例 210　汽车类使用 public 关键字定义汽车的行驶方法 332
实例 211　使用 private 关键字定义汽车的颜色属性 333
实例 212　使用 protected 关键字定义汽车的保修年限 334
实例 213　苹果子类继承水果父类 336
实例 214　使用 parent 关键字调用父类的方法 338
实例 215　苹果子类中覆盖水果父类中的方法 339
实例 216　美食抽象类 340
实例 217　学生类多重接口的实现 342
实例 218　通过继承实现多态 344
实例 219　通过接口实现多态 346
实例 220　使用 final 关键字防止类被继承 347
实例 221　使用 static 关键字定义类的静态成员 349
实例 222　使用 clone 关键字实现对象的克隆 351
实例 223　使用__set()方法为类中未经定义的属性赋值 352
实例 224　使用__get()方法获取未声明属性的名称 355
实例 225　使用__call()方法打印类中未定义方法的信息 356
实例 226　使用__toString()方法将类的实例转化为字符串 357
实例 227　使用__isset()方法提示未定义属性信息 359
实例 228　使用单例模式制作数据库管理类 360
实例 229　使用策略模式打印客户端浏览器类型 363
实例 230　使用工厂模式设置用户访问权限 365

第 13 章　PDO 数据库抽象层 369

实例 231　连接 MySQL 数据库 370
实例 232　连接 MS SQL Server 数据库 371
实例 233　连接 Oracle 数据库 372
实例 234　通过 PDO 向数据库中添加数据 373



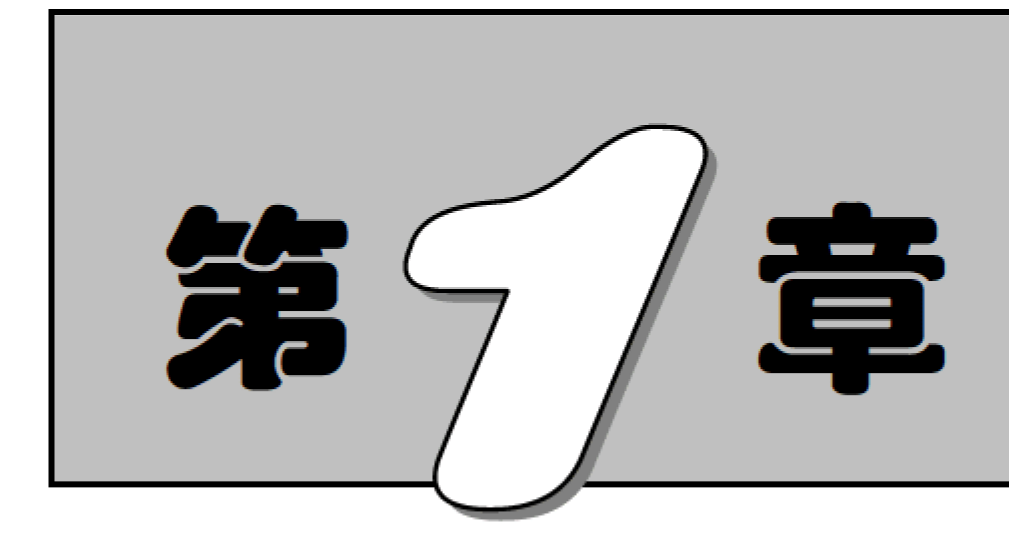
实例 235　通过 PDO 浏览数据库中的数据 374
实例 236　通过 PDO 更新数据库中的数据 375
实例 237　浏览客户留言 378
实例 238　明日书店会员注册 379
实例 239　添加留言信息 381
实例 240　查询留言 382

第 14 章　Smarty 模板 384
实例 241　封装 Smarty 模板的配置方法 385
实例 242　Smarty 模板中日期、时间的格式化输出 387
实例 243　Smarty 模板中的编码 388
实例 244　Smarty 模板制作日期、时间选择器 390
实例 245　通过 Section 循环输出数据 393
实例 246　开启网站注册页面的缓存 395
实例 247　通过配置文件定义变量 397
实例 248　Smarty+ADODB 完成数据的分页显示 399
实例 249　Smarty 模板中 truncate()方法截取字符串 401
实例 250　Register_function()方法注册模板函数 403
实例 251　Smarty 模板中的关键字描红技术 406
实例 252　Smarty 模板中生成数字验证码 408

第 15 章　ThinkPHP 框架 410
实例 253　用户信息的查询、更新和删除 411
实例 254　用户登录和数据的分页输出 415
实例 255　通过验证码类和分页类完成用户登录和分页输出 419
实例 256　通过 ThinkPHP 中的扩展类生成中文验证码 423
实例 257　可以传递查询条件的分页 424
实例 258　应用 ThinkPHP 中的扩展类上传文件 427

第 16 章　PHP 的字符编码 431
实例 259　设计 GB2312 编码格式的网页 432
实例 260　设计 GBK 编码格式的网页 433
实例 261　设计 UTF-8 编码格式的网页 434
实例 262　通过 iconv()函数实现编码格式的转换 435
实例 263　避免截取超长文本时出现乱码 436
实例 264　对输出的数据进行编码格式转换 438
实例 265　论坛中控制帖子标题输出的长度 439

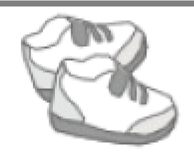




PHP 基础

本章读者可以学到如下实例：

- ⏭ 实例 001　获取当前执行文件的名称
- ⏭ 实例 002　计算一个圆形的面积
- ⏭ 实例 003　当数字遇到字符串
- ⏭ 实例 004　通过转义字符输出特殊字符串
- ⏭ 实例 005　通过 PHP5 新型字符串动态输出 JavaScript 代码
- ⏭ 实例 006　比较两个时间戳的大小
- ⏭ 实例 007　判断数字的奇偶性
- ⏭ 实例 008　通过逻辑运算符判断用户的权限
- ⏭ 实例 009　获取 2000～2020 年中所有的闰年
- ⏭ 实例 010　自定义数字加密算法
- ⏭ 实例 011　随机输出字符串
- ⏭ 实例 012　在页面中打印服务器时间
- ⏭ 实例 013　区分单引号和双引号
- ⏭ 实例 014　前置运算符和后置运算符的区别
- ⏭ 实例 015　使用可变变量输出“I Like PHP!”

实例001　获取当前执行文件的名称

（实例位置：配套资源\SL\01\001　视频位置：配套资源\SP\01\001）



实例说明

很多时候用户需要编写包含文件路径及文件名称的代码，如果通过目录去查找，未免有些麻烦，这时可以使用__FILE__预定义函数。本实例通过__FILE__预定义常量获取目标文件的路径及文件名称并在网页上进行打印，运行结果如图 1.1 所示。

D:\AppServ\www\sl\01\001\index.php

图 1.1　输出当前文件路径和名称

实现过程

创建 index.php 文件，并通过 echo 语句输出__FILE__预定义常量。其代码如下：

```
<?php
echo __FILE__;                          //获取文件的完整路径
?>
```

技术要点

系统预定义常量和用户自定义常量在使用上没有差别。大多数的预定义常量的执行结果都是服务器的相关信息（如版本号、路径、错误参数等），所以程序员很少将此函数用于网站前台的开发，如果被别有用心的人知道了这些信息，会严重威胁到服务器的安全。使用此函数的语法说明如下：

__FILE__预定义常量：文件的完整路径和文件名。如果用在包含文件中，则返回包含文件名。自 PHP 4.0.2 起，__FILE__总是包含一个绝对路径，而在此之前的版本有时会包含一个相对路径。

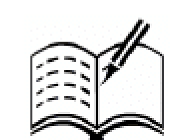
多学两招：

PHP 中可以使用预定义常量获取 PHP 中的信息。常用的预定义常量如表 1.1 所示。

表 1.1　PHP 的预定义常量

常　量　名	功　　能
__FILE__	默认常量，PHP 程序文件名
__LINE__	默认常量，PHP 程序行数
PHP_VERSION	内建常量，PHP 程序的版本，如“3.0.8_dev”
PHP_OS	内建常量，执行 PHP 解析器的操作系统名称，如“Windows”
TRUE	这个常量是一个真值（true）
FALSE	这个常量是一个假值（false）
NULL	一个 null 值
E_ERROR	这个常量指到最近的错误处
E_WARNING	这个常量指到最近的警告处





续表

常　量　名	功　　能
E_PARSE	这个常量指解析语法有潜在问题处
E_NOTICE	这个常量为发生不寻常但不一定是错误处

脚下留神：

__FILE__和__LINE__中的“__”是两条下划线，而不是一条“_”。表 1.1 中以 E_开头的预定义常量是 PHP 的错误调试部分。如果读者想详细了解，可参考 error_reporting()函数。

实例 002　计算一个圆形的面积

（实例位置：配套资源\SL\01\002　视频位置：配套资源\SP\01\002）

实例说明

常量是 PHP 编程基础的重要组成部分，其作用是定义一个不会改变的值。本实例通过计算圆的面积向用户说明常 8 量是如何定义和使用的，运行结果如图 1.2 所示。

半径为12个单位的圆的面积452.3893344

图 1.2　使用常量指定半径的值并计算圆的面积

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，通过 define()函数定义常量，将数值“3.1415926”定义常量名为 PI。然后，定义数值型变量，将圆的半径设定为 12 个单位。最后，通过 echo 语句输出圆面积。其代码如下：

```
<?php
    define("PI",3.1415926);                              //定义常量
    $r = 12;                                             //定义圆半径
    echo "半径为 12 个单位的圆的面积".PI*($r*$r);         //定义圆面积
?>
```

（2）将该文件存储于\MR\01\002\文件夹下，并命名为 index.php。运行结果如图 1.2 所示。

技术要点

PHP 中通常使用 define()函数来定义常量；使用 constant()函数动态获取常量值；使用 defined()函数判断一个常量是否已经定义；使用 get_defined_constants()函数获取所有当前已经定义的常量。本实例主要使用 define()函数来定义常量，该函数语法如下：

```
bool define ( string name, mixed value [, bool case_insensitive] );
```

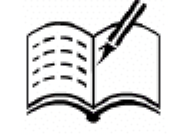


参数说明：

☑ string name：必选参数，常量名称，即标识符。

☑ mixed value：必选参数，常量的值。

☑ bool case_insensitive：可选参数，指定大小写敏感，设定为 true，表示不敏感。



Note

脚下留神：

用 define()定义的常量一旦定义就不能改变或取消。

实例 003　当数字遇到字符串

（实例位置：配套资源\SL\01\003　视频位置：配套资源\SP\01\003）

实例说明

在 PHP 手册中经常会看到一些返回值为 Boolean 的函数，函数说明的讲解可能是当符合条件时，返回值为 1，否则返回值为 0。按照我们正常的思维返回值应该是 true 和 false，因为它们才是真正的 Boolean 类型，其实这些都是类型转换的应用。本实例介绍 PHP 中的数据类型的知识，展示数据是如何进行类型转换的，运行结果如图 1.3 所示。

自动类型转换：
20+我是字符串型数据=20
强制类型转换：
20+我是字符串型数据=20

图 1.3　当数字遇到了字符串

实现过程

具体步骤如下：

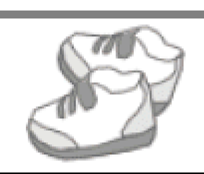
（1）创建 PHP 脚本，并通过 echo 语句输出想要转换的变量。其代码如下：

```
<?php
        $a = 20;                        //整型数据
        $b = "我是字符串型数据";         //字符串类型数据
        $e = $a + $b;
        $f = $b + $a;
        echo "自动类型转换：<br>";
        echo '20+我是字符串型数据='.$e."<br>";
        echo "强制类型转换：<br>";
        echo '20+我是字符串型数据='.(string)$e."<br>";
?>
```

（2）将文件存储于\MR\01\003\文件夹下，并命名为 index.php。运行结果如图 1.3 所示。

技术要点

虽然 PHP 是弱类型语言，但有时仍然需要用到类型转换。PHP 中的类型转换和 C 语言一样，非常简单。本实例的关键点是运用类型转换的特点进行输出。





多学两招：

（1）转换成整型。在将非整型数据转换成整型时，方法是在变量前使用“(integer)”或“(int)”。转换规则如下：

浮点型转换成整型。小数点后的数将被舍弃；如果浮点数超出整数取值范围，那么将无法得到有效的整型结果，结果可能是 0 或者整型的最小负数。

布尔型转换成整型。那么 true 值将转换为 1，false 值将转换为 0。

字符串型转换为整型。将会对字符串左侧的第一位进行判断。如果第一位是数字，则从第一位开始将读取到的数字转换成整型；如果第一位不是数字，则结果为 0。

（2）转换成浮点型。方法是在变量前使用“(float)”。转换规则如下：

整型转换为浮点型。其结果不会发生变化。

布尔型转换为浮点型。同样，true 值将转换为 1，false 值将转换为 0。

字符串型转换为浮点型。如果字符串中包含小数点“.”或科学记数法的“e”或者“E”中的任何一个字符，则字符串被当作浮点型处理，否则，被视为整型。

（3）转换成字符串型。方法是在变量前使用“(string)”。转换规则如下：

整型或者浮点型转换成字符串型。转换结果为其数值。

布尔型转换为字符串型。true 值将转换为字符串“1”，false 值将转换为空字符串。

对象或数组型转换成字符串型。那么转换结果为字符串对象或字符串数组。

资源型转换成字符串型。转换结果为一个类似“Resource id #”的字符串。在“#”之后是 PHP 在运行时分配给该资源的标识代号。

（4）转换成布尔型。方法是在变量前使用“(boolean)”或“(bool)”。因为布尔型只包含两个值：true 和 false，所以其转换规则包括两个方面：

第一方面，转换结果为 false 的情况。

整型或者浮点型数 0。

空字符串和字符串“0”。

没有任何元素的空数组。

没有任何元素的对象。

特殊类型“NULL”。

第二方面，转换结果为 true 的情况。排除在第一方面说明的情况，其他转换结果都为 true。

（5）转换成数组。方法是在变量前使用“(array)”。转换规则是：将非数组型转换成与原变量数据类型相同的数组，数组中只有一个元素。

（6）转换成对象。方法是在变量前使用“(object)”。转换规则是：将非对象型转换成一个新的对象，其中名为 scalar 的成员变量将包含原变量的值。

实例 004　通过转义字符输出特殊字符串

（实例位置：配套资源\SL\01\004　视频位置：配套资源\SP\01\004）

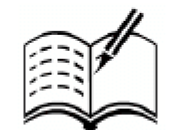
实例说明

在定义长字符串变量时，往往字符串本身包含一些特殊字符，如字符串本身包含双引

号（“”），这时就需要转义字符对特殊字符进行转义。本实例通过转义字符“\”转义经特殊字符双引号（“”）包含的字符串，运行结果如图 1.4 所示。



图 1.4　使用转义字符输出特殊字符串



实现过程

具体步骤如下：

（1）创建 PHP 脚本文件，并通过 echo 语句输出经转义字符转义的特殊字符串。其代码如下：

```
<?php
    echo "\"我喜欢 PHP！ \"";   //输出转义字符转义的字符串
?>
```

（2）将该文件存储于\MR\01\004\文件夹下，并命名为 index.php。运行结果如图 1.4 所示。

技术要点

本实例的关键点是转义字符“\”的使用。它除了可以做转义字符外，还有其他一些功能，可以将一些不可打印字符显示出来，如\a、\b、\e、\f、\n、\r 和\t 等。

指点迷津：
转义字符的功能只有一个，就是将特殊字符转义成普通字符。

实例 005　通过 PHP5 新型字符串动态输出 JavaScript 代码

（实例位置：配套资源\SL\01\005　视频位置：配套资源\SP\01\005）

实例说明

JavaScript 语言是一门功能强大的客户端脚本语言，也是一门跨平台语言。PHP 支持使用 JavaScript 编码。本实例通过 PHP5 新型字符串动态输出 JavaScript 代码，运行结果如图 1.5 所示。

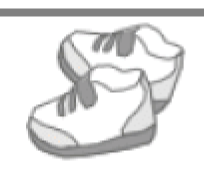


图 1.5　动态输出 JavaScript 代码

实现过程

创建 index.php 文件，并通过 PHP5 新型字符串实现实例。其代码如下：

```
<?php
$str=<<<mark                                            //PHP5 新型字符串开始部分
<script language="javascript" type="text/javascript">   //JavaScript 代码
        alert("欢迎进入 PHP 编程世界！！！ ");            //输出提示
</script>                                               //结束标记
mark;                                                   //新型字符串结束标记
echo $str;                                              //输出字符串
?>
```



技术要点

PHP5 新型字符串的使用是本实例的关键。PHP5 这种新型的字符串是以“<<<”开始，后面紧跟着字符串开始标记，之后为字符串的内容，最后以标记加分号（本实例以“mark”作为结束标记）结束。本函数一般应用于对 HTML 代码的格式输出。



Note

实例 006　比较两个时间戳的大小

（实例位置：配套资源\SL\01\006　视频位置：配套资源\SP\01\006）

实例说明

在一些程序中经常需要将两个时间进行比较，但是由于时间是由年、月、日、时、分、秒组成的，比较起来很不方便，这时可以把时间转换成时间戳来进行比较。本实例通过 date()函数、strtotime()函数和 ceil()函数实现比较两个时间戳的大小，运行结果如图 1.6 所示。

1305012224 输出日期2011-05-10 15:23:44

1304524800 输出日期2011-05-05 00:00:00

距2011年5月5号已过去6天

图 1.6　比较两个时间戳的大小

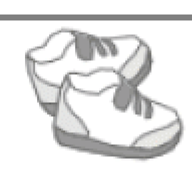
实现过程

具体步骤如下：

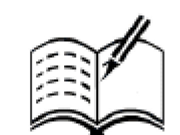
（1）创建 PHP 脚本文件。首先对 PHP 语言中的时区进行设置，将时区更改为中国上海时区。然后通过 strtotime()函数获取当前时间的 UNIX 时间戳，并与指定的时间戳进行算术运算。最后，将时间戳运算结果转换为天数，通过 ceil()函数进行取整运算并输出结果。其代码如下：

```
<?php
    date_default_timezone_set("Asia/ShangHai");                  //将格林威治时间设置为本地时间
    $a = strtotime("now");                                       //取得当前时间戳
    $b = strtotime("05 May 2011");                               //取得 2011 年 5 月 5 号的时间戳
    echo $a."\n";
    echo "输出日期".date("Y-m-d H:i:s",$a)."<br><br>";           //取得以$a 为时间点的时间
    echo $b."\n";
    echo "输出日期".date("Y-m-d H:i:s",$b)."<br><br>";           //取得以$b 为时间点的时间
    $c = ceil(($a - $b)/(3600*24));                              //计算相差天数
    echo "距 2011 年 5 月 5 号已过去".$c."天";
?>
```

（2）将该文件存储于\MR\01\006 文件夹下，并命名为 index.php。运行结果如图 1.6 所示。







Note

多学两招：

有效的时间戳通常从 Fri, 13 Dec 1901 20:45:54 GMT 到 Tue, 19 Jan 2038 03:14:07 GMT（对应于 32 位有符号整数的最小值和最大值）。不是所有的平台都支持负的时间戳，那么日记范围就被限制为不能早于 UNIX 纪元。这意味着在 1970 年 1 月 1 日之前的日期将不能用在 Windows、某些 Linux 版本以及几个其他的操作系统中。

技术要点

本实例主要是通过 strtotime()函数将时间转换为 UNIX 时间戳，并进行相关取整运算而实现的。其中涉及相关函数语法如下。

（1）strtotime()函数：本函数预期接受一个包含英文日期格式的字符串并尝试将其解析为 UNIX 时间戳。

```
int strtotime( string time [, int now]);
```

该函数有两个参数。如果参数 time 的格式是绝对时间，则 now 参数不起作用；如果参数 time 的格式是相对时间，其对应的时间就是参数 now 来提供的，当没有提供参数 now 时，对应的时间就为当前时间。如果解析失败，则返回 false。在 PHP5.1.0 之前的版本中，本函数在失败时返回-1。

（2）ceil()函数：返回不小于 value 的下一个整数。

```
float ceil ( float value);
```

脚下留神：

在 UNIX 系统中，日期与时间表示为自 1970 年 1 月 1 日零点起到当前时刻的秒数，这种时间称为 UNIX 时间戳，以 32 位二进制表示。其中，1970 年 1 月 1 日零点称为 UNIX 世纪元。UNIX 时间戳提供了一种统一、简洁的时间表示方式，在不同的操作系统中均支持这种时间表示方式，同一时间在 UNIX 和 Windows 中均以相同的 UNIX 时间戳表示，所以不需要在不同的系统中进行转换。同时，UNIX 时间戳是一个时间差，与时区没有关系，无论当前 PHP 中使用的是何种时区，其 UNIX 时间戳都是唯一的。

实例 007　判断数字的奇偶性

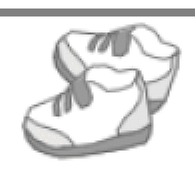
（实例位置：配套资源\SL\01\007　视频位置：配套资源\SP\01\007）

实例说明

条件运算符是进行或真或假运算的。本实例通过三元运算来演示如何用三元运算符判断数字的奇偶性，运行结果如图 1.7 所示。

0是偶数 1是奇数 2是偶数 3是奇数
5是奇数 6是偶数 7是奇数 8是偶数 9是奇数

图 1.7　用三元运算符判断数字的奇偶性



Note

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，通过 for 循环语句循环输出 0～9 的 10 个数字。然后，将得到的 10 个数字分别与 2 做求余运算并将运算结果作为三元运算的条件进行判断。最后，将判断的结果利用 echo()语句输出出来。其代码如下：

```
<?php
  //做小于 10 的循环
      for($a = 0;$a < 10;$a++){
  //用三元运算输出，如果没有余数则是偶数，否则为奇数
          echo    $a % 2 == 0 ?    $a."是偶数"."\n" : $a."是奇数"."\n";
}
?>
```

（2）将该文件存储于\MR\01\007\文件夹下，并命名为 index.php。运行结果如图 1.7 所示。

技术要点

本实例通过三元运算符判断数字的奇偶性，并且判断数字奇偶性时使用了求余运算符。当期望数字与 2 做除法运算时，如果存在余数（即不能被 2 整除），为奇数，否则为偶数。三元运算符的语法如下：

```
(expr) ? {statement1;} :   {statement2;}
```

三元运算符与 if…else…函数实现的功能完全相同。

实例 008　通过逻辑运算符判断用户的权限

（实例位置：配套资源\SL\01\008　视频位置：配套资源\SP\01\008）

实例说明

逻辑运算符往往作为 if 等语句的条件出现，它有很多种而且功能也各不相同。本实例介绍逻辑运算符的语法等相关知识，并通过逻辑运算符来演示用户是否具有后台管理权限，运行结果如图 1.8 和图 1.9 所示。

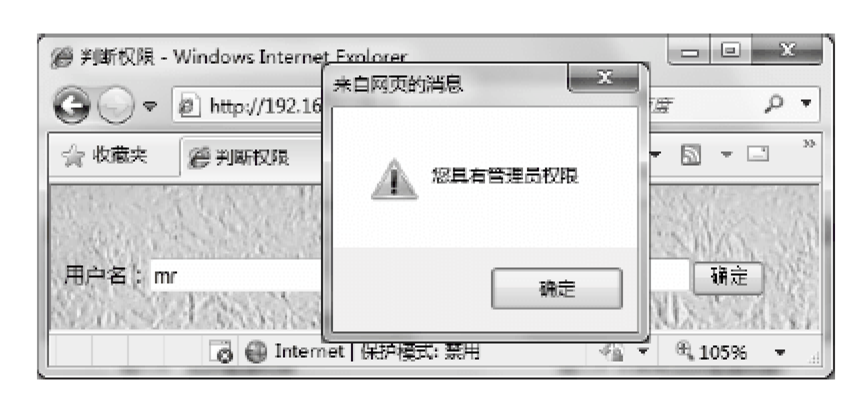


图 1.8　后台管理权限

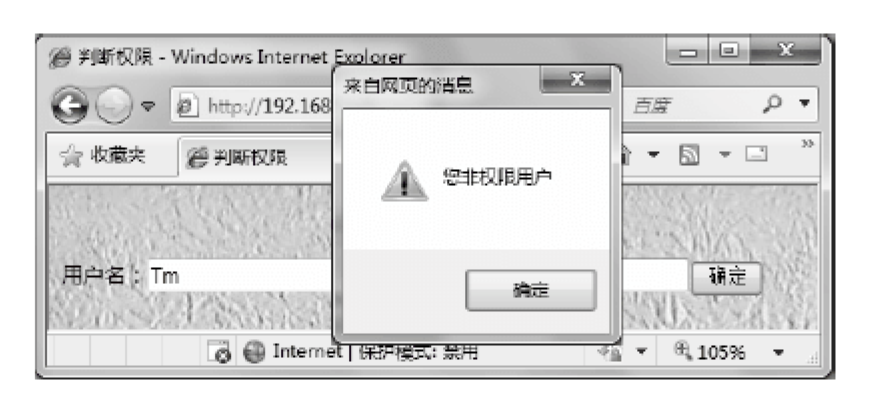
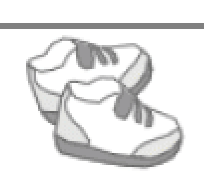


图 1.9　不具有后台管理权限

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，编写<form>表单，并将表单中 method 方法定义为





POST。然后，当单击按钮时，将 POST 方法接收的用户名和密码进行逻辑与运算。当 if 语句的结果为 true 时输出“您具有管理员权限”的提示，否则输出“您非权限用户”的提示。其代码如下：

```
<?php
    echo "<form action="method='post'>";                                  //输出表单
    echo "用户名：<input type='text' name='text'><br>";
    echo "密  码：<input type='password' name='pwd'>";
    echo "<input type='submit' name='sub' value='确定'>";
    echo "</form>";
    if(isset($_POST['sub'])){                                              //通过 post 方式提交
        if($_POST['text'] == "mr" && $_POST['pwd'] == "mrsoft"){          //验证用户名密码
            echo "<script>alert('您具有管理员权限');</script>";             //符合条件输出提示
        }else{
            echo "<script>alert('您非权限用户');</script>";                 //不符合条件输出提示
        }
    }
?>
```

（2）将该文件存储于\MR\01\008\文件夹下，并命名为 index.php。运行结果如图 1.8 和图 1.9 所示。

多学两招：

如果在涉及到运算符的优先级时不想总查找优先级表而本身又记忆不清时，建议使用小括号把想要先运算的表达式括起来，这样问题就迎刃而解了。

技术要点

本实例主要是通过逻辑运算符&&（逻辑与）将从表单中利用 POST 方法取得的由文本框输入的用户名和密码联系起来，用户名和密码必须同时为真时，if 语句的结果才会为真，反之为假。

实例 009　获取 2000～2020 年中所有的闰年

（实例位置：配套资源\SL\01\009　视频位置：配套资源\SP\01\009）

实例说明

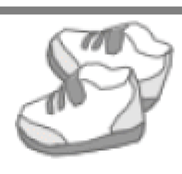
在 PHP 语言中为了提升编程效率设置了一些自增自减运算符，这种运算符在循环语句中得到广泛应用。本实例简要介绍了自增自减运算符并通过自增自减运算符修饰的变量来演示它的应用，运行结果如图 1.10 所示。

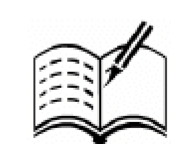


图 1.10　获取 2000～2020 年中所有的闰年

实现过程

创建 index.php 文件。首先，利用 for 循环定义循环变量，将变量的初始值设置为 2000，





最大值小于 2020，并且要求变量做自增运算。然后，设置循环变量与 4 做求余运算。最后，把符合条件的数值通过 echo 语句输出。其代码如下：

```
<?php
  if($_POST[sub]){                              //通过 post 接收参数
        for($a = 2000;$a <= 2020;$a++){         //通过循环定义循环变量
              if($a % 4 == 0){                  //将循环变量与 4 做求余运算
                    echo $a."  ";     //输出循环变量
              }
        }
  }
?>
```

技术要点

递增或递减运算符有两种使用方法：一种是先将变量增加或减少 1 后再将值赋给原变量，称为前置递增或递减运算符（也称前置自增自减运算符）；另一种是将运算符放在变量后面，即先返回变量的当前值，然后将变量的当前值增加或减少 1，称为后置递增或递减运算符（后置自增自减运算符）。本实例将递增递减运算符应用到 for 循环的变量之中，利用变量的变化完成期望次数的循环。

实例 010　自定义数字加密算法

（实例位置：配套资源\SL\01\010　视频位置：配套资源\SP\01\010）

实例说明

位运算符也是 PHP 运算符中不可或缺的一部分。本实例通过位运算符实现对数字进行加密，运行结果如图 1.11 和图 1.12 所示。

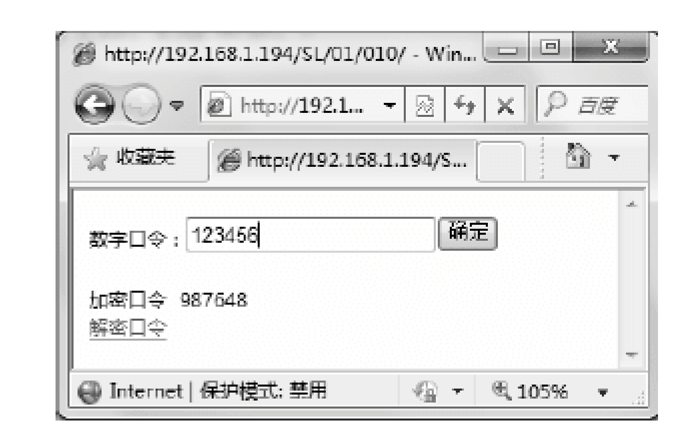


图 1.11　使用位运算符对数字进行加密

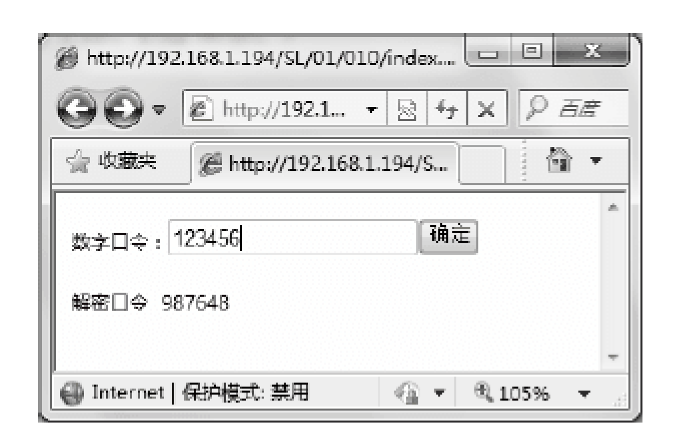


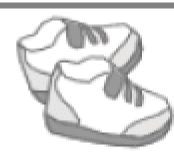
图 1.12　使用位运算符对数字进行解密

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，编写<form>表单并定义常量，将数值 3.1415926 赋给常量 PI。然后，定义自定义函数，在函数体内部将文本框信息与常量 PI 做移位运算并返回结果。最后，通过提交数据和单击按钮实现输出加密、解密算法的结果。其代码如下：

```
<?php                                                   //定义表单
      echo "<form action='' method='post'>";
      echo "数字口令：<input name='text' type='text'>";
```

```
        echo "<input type='submit' name='sub' value='确定'>";
        echo "</form>";
        define("PI",3.1415926);                              //定义常量
        function Encrypt($str){                              //自定义加密算法
            return $str = $str << PI;
        }
        function Decrypt($str){                              //自定义解密算法
            return $str = $str >> PI;
        }
        if($_POST[sub]){                                     //通过 post 方式传参数
            echo "加密口令  ".Encrypt($_POST[text])."<br>";     //输出口令
            $_SESSION[pwd] = Encrypt($_POST[text]);          //将口令保存在 SESSION 中
?>
            <a href='index.php?pwd=1'>解密口令</a>             //超链接
<?php
        }
        if(isset($_GET[pwd])){                               //判断地址栏是否存在此参数
            echo "解密口令  ".Decrypt($_SESSION[pwd]);          //输出解密结果
        }
?>
```

（2）将该文件存储于\MR\01\010\文件夹下，并命名为 index.php。运行结果如图 1.11 和图 1.12 所示。

多学两招：

位运算符允许对整型数据中指定的位进行置位。如果左右参数都是字符串，则位运算符将操作字符的 ASCII 值。

技术要点

位运算符是指对二进制位从低到高位对齐后进行运算。在 PHP 中的位运算符如表 1.2 所示。本实例中，利用自定义函数，将文本框信息与定义的常量做指定的位运算，并返回计算结果，从而将文本框信息进行加密或解密处理。

表 1.2　位运算符

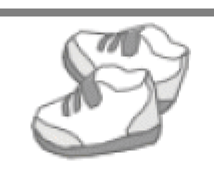
符　号	作　用	举　例	符　号	作　用	举　例
&	按位与	$m&$n	~	按位取反	$m~$n
|	按位或	$m\|$n	<<	向左移位	$m<<$n
^	按位异或	$m^$n	>>	向右移位	$m>>$n

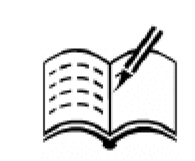
实例 011　随机输出字符串

（实例位置：配套资源\SL\01\011　视频位置：配套资源\SP\01\011）

实例说明

在应用字符串时往往需要将两个字符串进行连接，这时可以使用字符串运算符。本实





例通过字符串运算符打印随机组合生日祝福语，运行结果如图 1.13 所示。

生日快乐祝你万事如意

图 1.13　打印随机组合生日祝福语

实现过程

创建 index.php 文件。首先，定义两个数组，将字符串信息保存在数组之中。然后，利用 rand 函数随机取得数组中的两条信息保存在变量中。最后，通过 echo 语句输出用连接运算符连接的变量。其代码如下：

```
<?php
  $arr = array("生日快乐","今天是你的出生日","同学们为你许愿");                //定义数组
  $array = array("祝你万事如意","祝你生日快乐","祝你福如东海长流水寿比南山不老松");//定义
数组
        $rand = rand(0,2);                                                    //定义随机数
        echo $arr[$rand].$array[$rand];                                       //输出字符串
?>
```

脚下留神：

用过 C 或 Java 的读者应该注意，这里的“+”号只作为算术运算符使用，而不能作为字符串运算符。

技术要点

字符串可以用“.”（点）字符串连接符连接，它可以把两个或两个以上的字符串连接成一个新的字符串。字符串的连接有两种形式：第一种是连接运算符“.”；第二种是连接赋值运算符“.=”。在实际的程序开发中，两种连接方式都非常常用。

实例 012　在页面中打印服务器时间

（实例位置：配套资源\SL\01\012　视频位置：配套资源\SP\01\012）

实例说明

在用 PHP 编写的网站中，我们经常需要获取当前时间。例如，用户在什么时间登录的网站，黑客攻击网站是在什么时间等。本实例应用 PHP 标记通过 echo 语句和 date()函数来实现获取服务器时间，并进行打印，运行结果如图 1.14 所示。

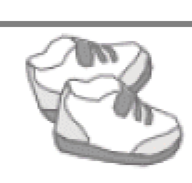
系统的当前时间是：2011-06-01 16:29:26

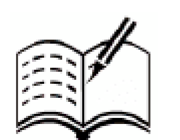
图 1.14　系统当前时间

实现过程

创建 index.php 文件，并通过 echo 语句输出 date()函数的返回值。其代码如下：

```
<?php
echo date('Y-m-s H:i:s');                    //输出当前时间
?>
```





技术要点

本实例主要是通过 date()函数获取系统的格林威治时间。其语法说明如下：

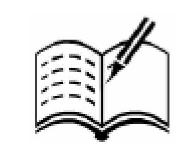
```
date(string format,int timestamp);
```

其中参数 format 指定日期和时间输出的格式。有关参数 format 指定的格式如表 1.3 所示。参数 timestamp 为可选参数，指定时间戳，如果没有指定时间戳，则使用本地时间 time()。

表 1.3　参数 format 的格式化选项

参　数	说　明
a	小写的上午和下午值，返回值 am 或 pm
A	大写的上午和下午值，返回值 AM 或 PM
B	Swatch Internet 标准时间，返回值 000～999
d	月份中的第几天，有前导零的 2 位数字，返回值 01～31
D	星期中的第几天，文本格式，3 个字母，返回值 Mon～Sun
F	月份，完整的文本格式，返回值 January～December
G	小时，12 小时格式，没有前导零，返回值 1～12
H	小时，24 小时格式，没有前导零，返回值 0～23
i	有前导零的分钟数，返回值 00～59
I	判断是否为夏令时，返回值如果是夏令时为 1，否则为 0
J	月份中的第几天，没有前导零，返回值 1～31
l	星期数，完整的文本格式，返回值 Sunday～Saturday
L	判断是否为闰年，返回值如果是闰年为 1，否则为 0
m	数字表示的月份，有前导零，返回值 01～12
M	3 个字母缩写表示的月份，返回值 Jan～Dec
n	数字表示的月份，没有前导零，返回值 1～12
o	与格林威治时间相差的小时数，如 0200
r	RFC 822 格式的日期，如 Thu，21 Dec 2000 16：01：07 +0200
s	秒数，有前导零，返回值 00～59
S	每月天数后面的英文后缀，两个字符，如 st、nd、rd 或 th。可以和 j 一起使用
t	指定月份所应有的天数
T	本机所在的时区
U	从 UNIX 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数
w	星期中的第几天，数字表示，返回值为 0～6
W	ISO-8601 格式年份中的第几周，每周从星期一开始
y	2 位数字表示的年份，返回值如 88 或 08
Y	4 位数字完整表示的年份，返回值如 1998、2008
z	年份中的第几天，返回值 0～366
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的，返回值：-43200～43200





Note

实例 013　区分单引号和双引号

（实例位置：配套资源\SL\01\013　视频位置：配套资源\SP\01\013）

实例说明

输出或定义字符串时离不开单引号和双引号的修饰，表面上看它们似乎没有什么不同，实际上它们是有区别的。本实例应用单引号与双引号操作符向用户说明如何区分单引号和双引号，运行结果如图 1.15 所示。

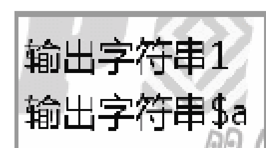


图 1.15　区分单引号和双引号

实现过程

创建 index.php 文件，并通过 echo 语句输出用单双引号修饰的字符串。其代码如下：

```
<?php
  $a = 1;
echo "输出字符串$a"."<br>";
echo '输出字符串$a';
?>
```

技术要点

本实例的关键点是输出单引号和双引号修饰的字符串以显示其区别。

使用双引号与单引号最大的区别是：双引号中所包含的变量会自动被替换成实际数值，而在单引号中包含的变量则按普通字符串输出。

实例 014　前置运算符和后置运算符的区别

（实例位置：配套资源\SL\01\014　视频位置：配套资源\SP\01\014）

实例说明

前置运算符与后置运算符，大多用在循环语句中。本实例应用自增自减运算符修饰的变量通过显示结果来说明前置运算符与后置运算符的区别，运行结果如图 1.16 所示。

```
----------后置加运算---------
0 1 2 3 4 5 6 7 8 9 10
----------前置加运算---------
1 2 3 4 5 6 7 8 9 10 11
----------前置减运算---------
9 8 7 6 5 4 3 2 1 0 -1
----------后置减运算---------
10 9 8 7 6 5 4 3 2 1 0
```

图 1.16　前置运算符与后置运算符区别





实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，通过 for 循环语句定义循环变量。然后，通过 echo 语句输出循环变量。其代码如下：

```
<?php
    $a = 0;                              //自定义变量并给定初始值
    echo "----------后置加运算---------<br>";
    for($b = 0;$b <= 10;$b++){           //使用 for 循环语句
        echo   $a++." ";                 //使用后置运算符
    }
    echo "<br>----------前置加运算---------<br>";
    $d = 0;
    for($b = 0;$b <= 10;$b++){
        echo ++$d." ";                   //使用前置运算符
    }
    echo "<br>----------前置减运算---------<br>";
    $f =10;
    for($b = 10;$b >= 0;$b--){
        echo --$f." ";                   //使用前置运算符
    }
    echo "<br>----------后置减运算---------<br>";
    $g =10;
    for($b = 10;$b >= 0;$b--){
        echo $g--." ";                   //使用后置运算符
    }
?>
```

（2）将该文件存储于\MR\01\014\文件夹下，并命名为 index.php。运行结果如图 1.16 所示。

技术要点

本实例的关键点是自增自减运算符的灵活运用。在使用前置减运算符时，要注意给定的初始值，否则会出现本例的问题，前置减运算输出负数。

实例 015　使用可变变量输出“I Like PHP!”

（实例位置：配套资源\SL\01\015　视频位置：配套资源\SP\01\015）

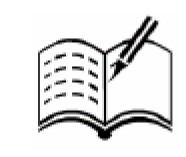
实例说明

变量包括预定义变量、可变变量等。本实例通过可变变量实现输出指定字符串，运行结果如图 1.17 所示。

I Like PHP!

图 1.17　使用可变变量输出





实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，定义一个字符串变量。然后，将第一个字符串变量的值前加个“$”作为第二个变量的变量名。最后，通过 echo 语句输出可变变量。其代码如下：

```
<?php
        $str_name = "str_name_1";              //定义变量
        $str_name_1 = "I Like PHP!";           //定义可变变量
        echo $$str_name;                       //输出可变变量
?>
```

（2）将该文件存储于\MR\01\015\文件夹下，并命名为 index.php。运行结果如图 1.17 所示。

技术要点

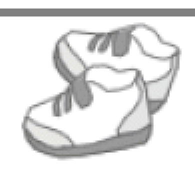
可变变量是一种独特的变量，它允许动态改变一个变量名称。工作原理是该变量的名称由另外一个变量的值来确定。实现过程是在变量的前面多加一个“$”。

第 2 章

函数

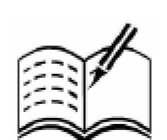
本章读者可以学到如下实例：

- ⏭ 实例 016　自定义函数过滤字符串
- ⏭ 实例 017　获取上传文件的后缀
- ⏭ 实例 018　统一上传文件名称的大小写
- ⏭ 实例 019　论坛内容的简单输出
- ⏭ 实例 020　日期、时间的格式化输出
- ⏭ 实例 021　倒计时
- ⏭ 实例 022　中文图像验证码
- ⏭ 实例 023　上传文件到服务器
- ⏭ 实例 024　读取整个文本文件中的内容
- ⏭ 实例 025　连接 MySQL 数据库服务器
- ⏭ 实例 026　选择指定的 MySQL 数据库
- ⏭ 实例 027　执行 SQL 语句
- ⏭ 实例 028　文本文件分页读取
- ⏭ 实例 029　查询关键字描红
- ⏭ 实例 030　分页显示图书信息
- ⏭ 实例 031　一般用户注册
- ⏭ 实例 032　带检测用户名的用户注册
- ⏭ 实例 033　分步用户注册



实例 016　自定义函数过滤字符串

（实例位置：配套资源\SL\02\016　视频位置：配套资源\SP\02\016）



实例说明

用户在论坛上发表留言或者回复时可能会遇到这种情况：单击“发布”按钮时，系统提示使用了禁用词语禁止发布。这是论坛的开发人员为了规范论坛而采用的过滤字符串的手段。本实例通过自定义函数实现过滤字符串，运行结果如图 2.1 所示。

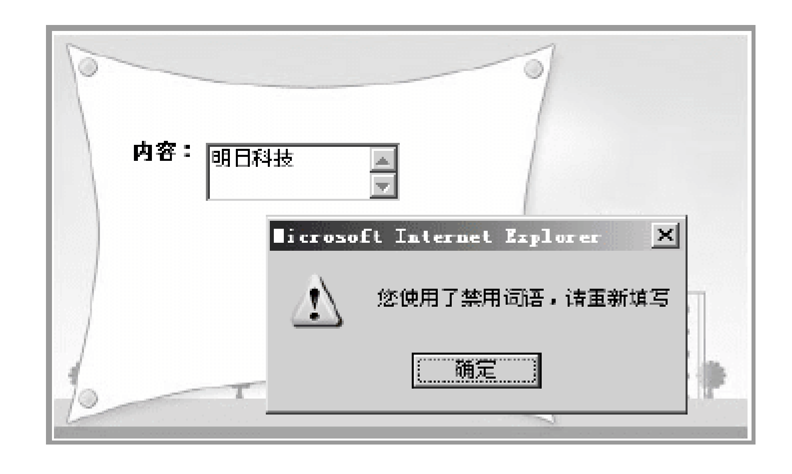


图 2.1　自定义函数过滤字符串

实现过程

具体步骤如下：

（1）创建名称为 index 的 PHP 文件，并自定义具有过滤功能的验证函数。首先，将禁用词语保存在数组中，并通过分割函数将数组拆分。然后，应用正则表达式函数对禁用词语进行判断。当用户输入内容中包含禁用词语时，会执行屏蔽操作。该过滤功能是通过当单击“提交”按钮时，调用 PHP 的自定义函数来实现的。最后，通过 JavaScript 输出提示信息。其代码如下：

```
<?php
 function str($str){                                                    //自定义函数
     $array = array('图书','明日科技','软件','编程词典','编程','词典');      //数组转换为字符串
     $repstr = implode($array);
     if(preg_match("/$str/",$repstr)){                          //正则表达式验证字符串
         echo "<script>alert('您使用了禁用词语，请重新填写');location.href='index.php'</script>";
     }else{
         echo "内容为：".$str;                                          //输出字符串
     }
 }
 if(!empty($_POST['sub'])){
     str($_POST['te']);                                                 //调用自定义函数
 }
?>
```

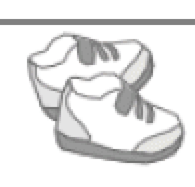
（2）将该文件存储于\MR\02\001\文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/001 浏览文件。

脚下留神：

在判断是否存在以 POST 方式传递的 URL 参数时，如果直接应用判断该传递参数是否存在，系统可能会给出未定义的警告提示。为了避免得到非预期结果，这里应用 empty()函数判断该传递参数是否存在。

技术要点

本实例通过创建自定义函数 str()对字符串进行过滤。如果出现指定的关键词，则给出



提示信息，并终止程序执行。自定义函数 str 的结构如下：

```
function str($str){
  $array = array('图书','明日科技','软件','编程词典','编程','词典');       //定义数组
  $repstr = implode($array);                                             //数组转换成字符串
  if(preg_match("/$str/",$repstr)){                                      //正则表达式验证字符串
      echo "<script>alert('您使用了禁用词语，请重新填写');location.href='index.php'</script>";
  }else{
      echo "内容为：".$str;                                               //输出数据
  }
}
```

多学两招：

本实例只是简单地介绍了字符串过滤的原理，真正的过滤程序由于数据量较大，需要将数据存储在数据库之中。如果读者感兴趣，可以将其与数据库中的数据结合进行使用。

实例 017　获取上传文件的后缀

（实例位置：配套资源\SL\02\017　视频位置：配套资源\SP\02\017）

实例说明

文件上传功能几乎是所有网站所必需的模块。然而，文件上传对于服务器来说具有很大风险，应该对文件大小、文件类型进行限制。本实例通过字符串函数 strrev()获取上传文件的后缀，运行结果如图 2.2 所示。

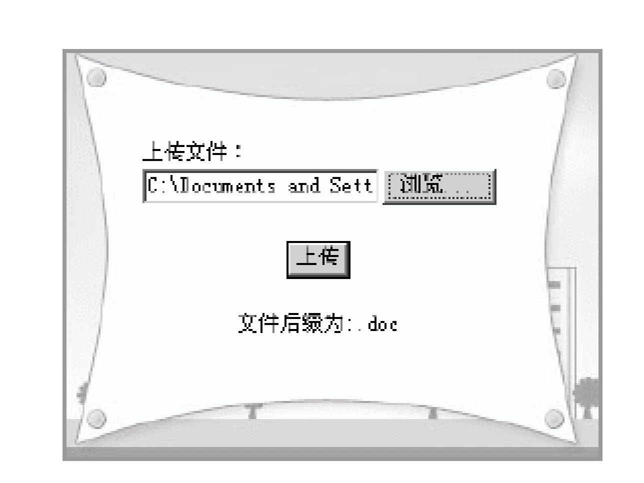


图 2.2　获取上传文件的后缀

实现过程

具体步骤如下：

（1）创建名称为 index 的 PHP 脚本文件。当用户单击“上传”按钮时，首先利用 post 方法接收文本框中的信息，其次利用反转函数将文本框信息数据进行反转，并利用字符串拆分函数 explode()以“.”作为分隔符进行分割，然后将分割后的结果保存在数组之中，最后将指定数据再次反转，输出已取得上传文件的后缀名称。其代码如下：

```
<?php
      f(!empty($_POST["sub"])){                    //通过 post 方式传递参数
            $a = strrev($_POST[text]);             //反转字符串
            $b = explode(".",$a);                  //以点号分割
            $c = strrev($b[0]);                    //反转字符串
            echo "文件后缀为:.$c";                  //输出文件后缀
      }
?>
```

（2）将该文件存储于\MR\02\002\文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/002 浏览文件。





技术要点

PHP 中字符串的逆序输出可以通过循环实现，但最简单的方式是通过函数 strrev()实现。strrev()函数用于将字符串反转。其语法如下：

```
string strrev ( string string );
```

参数 string string 为需要反转的字符串。

在本实例中，通过 strrev()函数对上传文件的名称进行反转输出，并通过 explode()函数以“.”为分隔符，对文件名称进行分割，然后再次应用 strrev()函数对数组中的第一个元素值进行返回输出，获取到的就是上传文件的后缀。

多学两招：

利用 strrev()函数达到检索字符串的目的相对较麻烦，可以使用正则表达式函数 preg_match()来完成。

实例 018　统一上传文件名称的大小写

（实例位置：配套资源\SL\02\018　视频位置：配套资源\SP\02\018）

实例说明

统一上传文件名称的大小写是十分必要的。例如，上传图片文件 img.jpg 和 Img.jpg，如果文件名称未经大小写统一则上传时可能就不会出现覆盖提示。本实例通过字符串函数 strtoupper()、strtolower()实现统一上传文件名称的大小写，运行结果如图 2.3 所示。

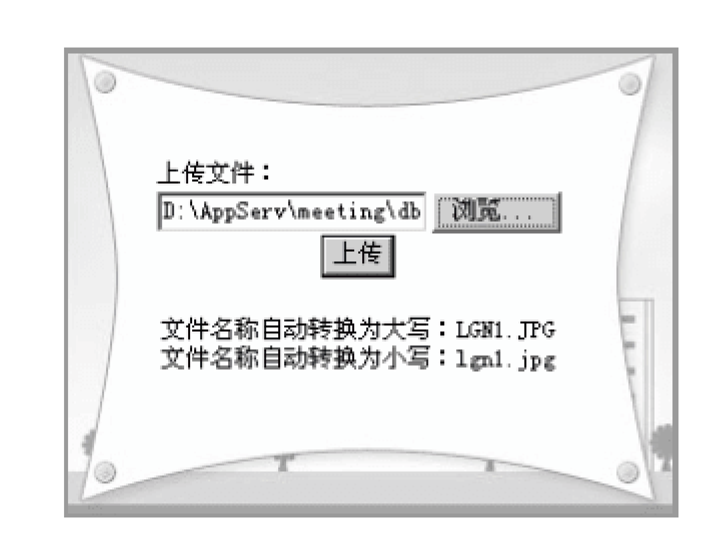


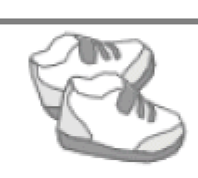
图 2.3　统一上传文件名称的大小写

实现过程

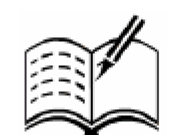
具体步骤如下：

（1）创建名称为 index 的 PHP 脚本文件。当用户单击“上传”按钮时，处理页分别利用转换小写和大写的函数 strtolower()、strtoupper()将接收到的文本框的信息进行对应的大小写转换，并输出转换结果。其代码如下：

```
<?php
    session_start();
     if(!empty($_POST["sub"])){                                    //通过 post 方式传递参数
          $a = strtoupper($_POST["text"]);                         //转换为大写
          echo "文件名称自动转换为大写：".$a."<br>";               //输出
          $b = strtolower($_POST["text"]);                         //转换为小写
          echo "文件名称自动转换为小写：".$b;                      //输出
     }
?>
```



（2）将该文件存储于\MR\02\003\文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/003 浏览文件。



Note

脚下留神：

session_start()开启会话是不能省略的。如果该语句缺少，则在程序运行时可能得不到预期结果，或在显示页面出现错误。

技术要点

实现字符串的大小写转换，主要通过 strtolower()和 strtoupper()函数。

（1）strtolower()：转换为小写。

```
string strtolower ( string str )
```

（2）strtoupper()：转换为大写。

```
string strtoupper ( string string )
```

参数 string 为需要转换的字符串。

多学两招：

在电子商务类网站的后台商品添加页面中，为了使商品的型号以统一的规格显示，经常会利用大小写转换函数，所以掌握这两个函数是很有必要的。

实例 019　论坛内容的简单输出

（实例位置：配套资源\SL\02\019　视频位置：配套资源\SP\02\019）

实例说明

开发论坛程序很重要的一点是转换空格符和回车符，因为不是所有的论坛用户都知晓 HTML 标签“ 和
的作用，大多数用户都喜欢用空格键和回车键对文本进行控制，这就需要程序员人为地做到统一。下面我们通过自定义函数定义转换方法实现论坛内容的简短输出，运行结果如图 2.4 所示。

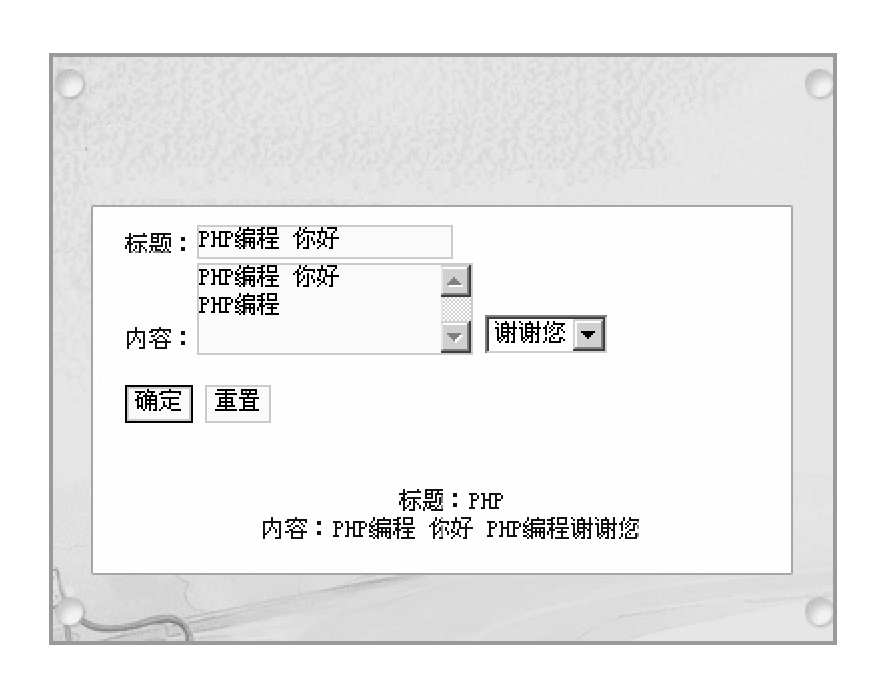


图 2.4　论坛内容的简短输出

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件，并通过创建自定义函数 str()对空格符和回车符进行替换。当用户单击“提交”按钮时，处理页首先对文本框是否存在内容进行判断，如果文本框内容不为空，则调用自定义函数并输出文本信息。其代码如下：






```
<?php
function str($str){                                        //自定义函数
  $str = preg_replace("/ /"," ",$str);                //替换空格符
  $str = preg_replace("/cha(13)/","<br>",$str);            //替换回车符
  echo "内容：".$str;                                       //输出
}
if(!empty($_POST["sub"])){                                 //通过 post 方式传递参数
//如果 text 为空或者文本域和下拉列表框同时为空
        if($_POST["text"] == "" || (($_POST["te"] == "") && ($_POST["check"] == 1))){
            echo "<script>alert('内容不能为空');location.href='index.php'</script>";   //输出提示
  }else{
        if(!empty($_POST["check"]) || $_POST["select"] != '1'){ //当下拉列表值不等于 1 时
            echo "标题：".$_POST["text"]."<br>";             //输出内容和下拉列表内容
            str($_POST[te].$_POST["check"]);
        }else{
            echo "标题：".$_POST["text"]."<br>";             //否则只输出文本域内容
            str($_POST["te"]);
        }
  }
}
?>
```

（2）将该文件存储于\MR\02\004\文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/004 浏览文件。

技术要点

本实例中通过自定义函数实现对字符串中的空格和回车符进行转换输出，在自定义函数中应用的是 preg_replace()函数，其结构如下：

```
function str($str){                                        //自定义函数
$str = preg_replace("/ /"," ",$str);                  //替换空格符
$str = preg_replace("/cha(13)/","<br>",$str);              //替换回车符
echo "内容：".$str;                                         //输出
}
```

多学两招：

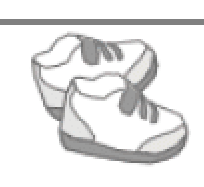
本例代码中 “$_POST["text"] == "" || (($_POST["te"] == "") && ($_POST["check"] == 1))” 是存在运算符优先级的。为了避免发生错误，笔者将先进行的运算人为加上了小括号。其中 $_POST["te"]==""可以用 empty($_POST["te"])代替。

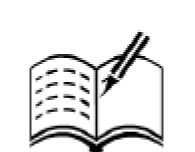
实例 020　日期、时间的格式化输出

（实例位置：配套资源\SL\02\020　视频位置：配套资源\SP\02\020）

实例说明

在 PHP 程序设计中，对日期、时间的格式化输出是经常用到的。本实例通过函数 date()





实现日期、时间的格式化输出，运行结果如图 2.5 所示。



图 2.5　日期、时间的格式化输出

实现过程

具体步骤如下：

（1）创建名称为 index 的 PHP 脚本。当用户单击“显示当前日期、时间”超链接时，利用 date()函数根据传入指定的参数格式而实现格式化输出。其代码如下：

```
<?php
date_default_timezone_set("Asia/ShangHai");              //设置时区
If($_GET[date] == 1){                                    //通过 get 方式取得参数
  cho date("Y-m-d H:i:s");                               //格式化输出时间和日期
}
?>
```

（2）将该文件存储于\MR\02\005 文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/005 浏览文件。

技术要点

date()函数用于获取系统的格林威治时间。其语法说明及相关参数介绍请参见实例 012。

多学两招：

获取当地时间时一定要先设置时区，否则 PHP 默认使用的是英国伦敦的零时区。而在我们本地使用的是北京时间，即东八区的时间，所以需要使用 date_default_timezone_set("Asia/ShangHai")设置时区。

实例 021　倒计时

（实例位置：配套资源\SL\02\021　视频位置：配套资源\SP\02\021）

实例说明

倒计时功能的实现，同样也是将时间戳做算术运算。本实例应用 strtotime()函数实现倒计时程序，运行结果如图 2.6 所示。

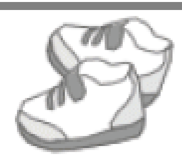
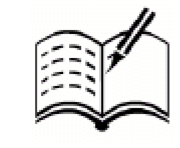




图 2.6　倒计时



实现过程

具体步骤如下：

（1）创建名称为 index 的 PHP 脚本文件，并设置当前时区为中国上海。首先，利用 time()函数取得当前时刻的时间戳并保存在变量中。然后，利用 strtotime()函数取得 2012 年 1 月 1 日的时间戳并保存在变量中。最后，将两个时间戳进行求差运算并将结果转换成天数后输出。其代码如下：

```
<?php
date_default_timezone_set("Asia/ShangHai");                    //设置时区
echo "当前时间：".date('Y 年 m 月 d 日  H 点 i 分 s 秒')."<br>";//输出当前日期
$time = time();                                                //取得当前时间戳
$time_r = strtotime("1 January 2012");                         //取得指定日期的时间戳
echo "距 2012 年元旦还有<b style='color:red;'>".ceil((($time_r - $time)/(3600*24)))."</b>天";//输出
?>
```

（2）将该文件存储于\MR\02\006\文件夹下，并在浏览器中输入 http://127.0.0.1/mr/02/006 浏览文件。

技术要点

PHP 中应用 strtotime 函数将任何英文文本的日期解析为 UNIX 时间戳，其值为相对于 now 参数给出的时间，如果没有提供此参数则用系统当前时间，此时与实例 142 中的 time()函数相同。strtotime 函数的语法如下：

```
int strtotime ( string time [, int now] );
```

该函数有两个参数。如果参数 time 的格式是绝对时间，则 now 参数不起作用；如果参数 time 的格式是相对时间，其对应的时间就是参数 now 来提供的，当没有提供参数 now 时，对应的时间就为当前时间。如果解析失败，则返回 false。在 PHP5.1.0 之前的版本中，本函数在失败时返回-1。

多学两招：

通过上面几个实例的学习，读者应该明白其实所有的时间比较到头来都是时间戳的比较，所以掌握时间戳是十分重要的。





Note

实例 022　中文图像验证码

（实例位置：配套资源\SL\02\022　视频位置：配套资源\SP\02\022）

实例说明

验证码不仅可以使用数字图像来生成，还可以使用中文图像来生成。为了防止用户通过恶意程序登录站点，提高网站的安全性，在本实例中将介绍如何通过中文图像来生成验证码，运行结果如图 2.7 所示。



图 2.7　中文图像验证码

实现过程

具体步骤如下：

（1）创建名称为 index.php 的用户登录页面，在该页面创建表单，在表单中添加用户名文本域、密码域、验证码文本域以及保存随机数的隐藏域。

（2）在 index.php 页面中编写 PHP 脚本，定义验证码所使用的中文字符串，并通过 rand()函数生成验证码，其关键代码请参考本实例的技术要点。

（3）本实例中沿用 JavaScript 脚本验证技术表单中元素进行验证的方法，fun.js 脚本文件同样存储于 js 文件夹中。

（4）创建后台验证文件 checkuser.php，并完成对用户提交的登录信息的验证。

脚下留神：

建议在应用循环前，定义$img 变量和$pic 变量为空值，否则在程序运行时，系统可能会报错。

技术要点

开发中文图像验证码，首先应该确定所使用的文字，并且将文字制作成图片，以数字进行命名。然后，将所使用的文字定义到数组中。最后，计算出数组中元素的长度值，并应用 rand()函数从这个长度值中随机取值，根据随机取值输出对应的中文图像，即验证码。其关键代码如下：



```
<?php
$str = array ("大", "更", "创", "天", "科", "客", "博", "技", "立", "新" );//定义中文字符串的数组
$word = count ( $str );                                   //计算出数组中元素个数
$img="";                                                  //定义$img 变量
$pic="";                                                  //定义$pic 变量
for($i = 0; $i < 4; $i ++) {        //执行 for 循环，以数组中元素个数为范围随机取出 4 个值
  $num = rand ( 0, $word - 1 ); //$word=$word*2-1               //随机取值
//输出随机取出的 4 个值对应的中文图片
  $img = $img . "<img src=' images/checkcode/" . $num . ".gif' width='16' height='16'>"; //显示随机
图片
  $pic = $pic . $str ["$num"];                             //将图片转换成数组中的文字
}
?>
```



多学两招：

在本实例中，应用 count()函数计算数组中元素的个数。注意，count()函数只适用于计算数组中元素的个数，如果将其应用到其他类型的数据中，那么返回值只有一个单元。如果要计算字符串的长度，那么就要使用 strlen()函数，切忌 strlen()函数不可对数组元素进行操作，虽然不返回错误信息，但是返回的结果也是不准确的。

实例 023　上传文件到服务器

（实例位置：配套资源\SL\02\023　视频位置：配套资源\SP\02\023）

实例说明

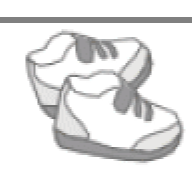
在开发网站的过程中，不但可以将文件上传到数据库中，还可以将文件上传到服务器，这种方式可以为网站中的数据库节省很多的空间，同时可以控制上传文件的大小、格式，而且读取服务器中的文件要比从数据库中读取方便很多。运行本实例，如图 2.8 所示，单击图中的“浏览”按钮，选择要上传的文件，然后单击“提交”按钮，如果上传成功，则在页面中提示“上传成功！”，否则给出错误提示信息。

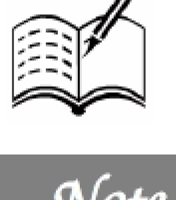


图 2.8　上传文件到服务器

实现过程

创建 index.php 文件。首先添加表单，并设置文件域和提交按钮。然后，使用 POST 方法设置 enctype="multipart/form-data"，并通过$_FILES 获取上传文件的相关信息。最后，





通过 move_upload_file()函数完成图片上传。其主要代码如下：

```
<?php
session_start();                                          //开启全局会话
if(!empty($_FILES["up_picture"]["name"])){                //判断上传内容是否为空
  if($_FILES['"up_picture"]["error"]>0){                  //判断文件是否可以上传到服务器
        echo "上传错误:";
        switch($_FILES["up_picture"]["error"]){           //如果上传错误，判断错误原因
              case 1:
                    echo "上传文件大小超出配置文件规定值";
              break;
              case 2:
                    echo "上传文件大小超出表单中约定值";
              break;
              case 3:
                    echo "上传文件不全";
              break;
              case 4:
                    echo "没有上传文件";
              break;
        }
  }else{
        if(!is_dir("./upfile/")){                                  //判断指定目录是否存在
              mkdir("./upfile/");                                  //创建目录
        }
        $path='./upfile/'.time().$_FILES["up_picture"]["name"];   //定义文件名称和存储位置
        if(is_uploaded_file($_FILES["up_picture"]["tmp_name"])){  //是否是 HTTP POST 上传
              if(!move_uploaded_file($_FILES["up_picture"]["tmp_name"],$path)){   //执行上传
                    echo "上传失败";
              }else{
                    echo "文件".time().$_FILES["up_picture"]["name"]."上传成功，大小为：".$_
FILES['up_picture']['size'];  //输出文件大小
              }
        }else{
              echo "上传文件".$_FILES["up_pictute"]["name"]."不合法！";
        }
  }
}
?>
```

技术要点

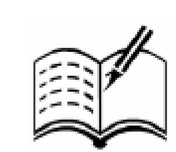
PHP 中可以应用 move_uploaded_file()函数实现文件上传。在执行文件上传之前，为了防止潜在的攻击对原本不能通过脚本交互的文件进行非法管理，可以先应用 is_uploaded_file()函数判断指定的文件是否是通过 HTTP POST 上传的，如果是则返回 true。

（1）is_uploaded_file()函数。

is_uploaded_file()函数，判断指定的文件是否是通过 HTTP POST 上传的。其语法如下：

```
bool is_uploaded_file ( string filename )
```





参数 filename 必须指定类似于$_FILES['filename']['tmp_name']的变量，不可以使用从客户端上传的文件名$_FILES['filename']['name']。

通过 is_uploaded_file()函数对上传文件进行判断，可以确保恶意的用户无法欺骗脚本去访问本不能访问的文件，如/etc/passwd。

（2）move_uploaded_file()函数。

move_uploaded_file()函数将文件上传到服务器中指定的位置。如果成功则返回 true，否则返回 false。其语法如下：

```
bool move_uploaded_file ( string filename, string destination )
```

参数 filename 指定上传文件的临时文件名，即$_FILES[tmp_name]；参数 destination 指定文件上传后保存的新路径和名称。

脚下留神：

如果参数 filename 不是合法的上传文件，则不会出现任何操作，move_uploaded_file()将返回 false。

如果参数 filename 是合法的上传文件，但出于某些原因无法移动，也不会出现任何操作，move_uploaded_file()将返回 false，此外还会发出一条警告。

实例 024　读取整个文本文件中的内容

（实例位置：配套资源\SL\02\024　视频位置：配套资源\SP\02\024）

实例说明

在开发网站的过程中，很多的服务条款、协议等都是以文本文件的形式存储的。如果要读取这些文件中的内容就需要应用到文件系统函数。也就是说，只有通过文件系统函数，才能读取其中的内容。本实例将介绍如何通过文件系统函数读取整个文本文件中的内容，运行结果如图 2.9 所示。

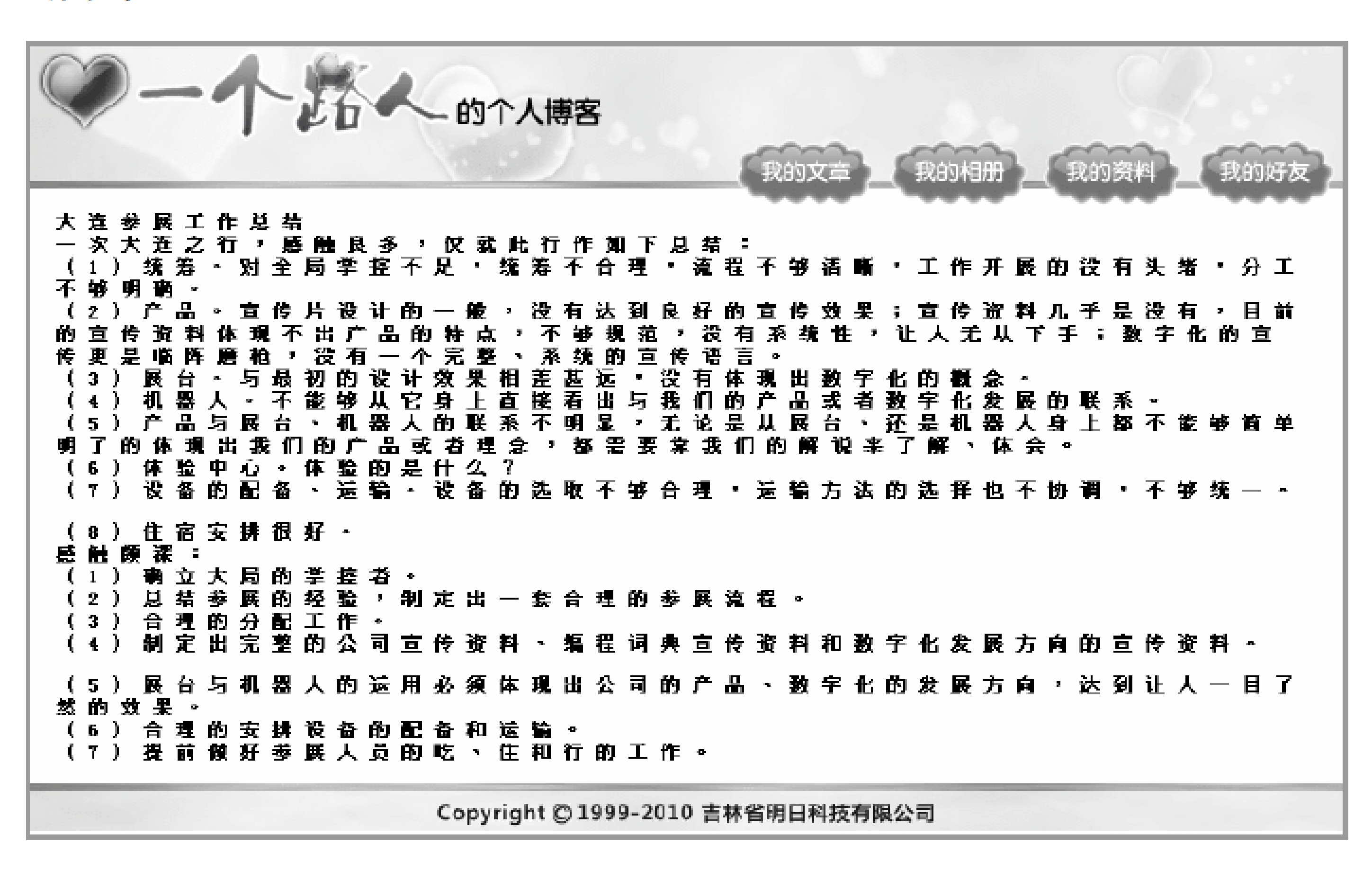
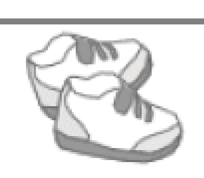
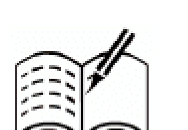


图 2.9　读取文本文件中的内容





Note

实现过程

具体步骤如下：

（1）创建 index.php 文件。

（2）在 index.php 文件中分别应用 file()和 file_get_contents()函数读取文本文件中的内容。其关键代码如下：

```
<!--  使用 file()函数读取 test1.txt 文件的内容  -->
<?php
$arr = file('test1.txt');                              //读取文件
foreach ($arr as $value) {                             //循环输出
    echo iconv("gb2312", "utf-8", $value) . "<br>";    //将读取的文字转换编码格式
}
?>
<!--  使用 file_get_contents()函数读取 count.txt 文件的内容  -->
<?php
$str = file_get_contents('test2.txt');                 //应用函数获取文本中的内容
echo iconv("gb2312", "utf-8", $str);                   //将读取的文字转换编码格式
?>
```

技术要点

（1）readfile()函数。

readfile()函数读取一个文件并写入到输出缓冲，成功则返回读取的字节数，失败则返回 false。其语法如下：

```
int readfile ( string filename [, bool use_include_path [, resource context]] )
```

参数 filename 指定读取的文件名称；参数 use_include_path 控制是否支持在 include_path 中搜索文件，如果支持，则将该值设置为 true；参数 context 是 PHP 5.0 新增的内容。

（2）file()函数。

file()函数将整个文件的内容读入到一个数组中。成功则返回数组，数组中的每个元素都是文件中对应的一行，包括换行符在内；失败则返回 false。其语法如下：

```
array file ( string filename [, int use_include_path [, resource context]] )
```

file()函数与 readfile()函数相同，唯一区别是该函数返回值是数组。

（3）file_get_contents()函数。

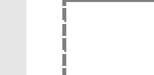
file_get_contents()函数将文件内容读入一个字符串。如果有 offset 和 maxlen 参数，将在参数 offset 所指定的位置开始读取长度为 maxlen 的内容。如果失败，则返回 false。其语法如下：

```
string file_get_contents ( string filename [, bool use_include_path [, resource context [, int offset [, int maxlen]]]] )
```

参数 filename 指定读取的文件名称；参数 use_include_path 控制是否支持在 include_path



中搜索文件，如果支持，则将该值设置为 true。



多学两招：

在通过 readfile()、file()和 file_get_contents()函数读取整个文件中的内容时，不需要通过 fopen()函数打开文件，也不需要使用 fclose()函数关闭文件。

但是，在读取一个字符、一行字符或任意长度的字符串时，必须应用 fopen()函数打开文件后才能进行读取，在读取完成后还要应用 fclose()函数关闭文件。

实例 025　连接 MySQL 数据库服务器

（实例位置：配套资源\SL\02\025　视频位置：配套资源\SP\02\025）

实例说明

通过 PHP 语言连接 MySQL 数据库与在 cmd 命令提示符下操作数据库有所不同。例如，设置页面的编码格式为“GBK”。在 MySQL 命令提示符下只需输入“set names GBK”即可，而在 PHP 中需使用函数“mysql_query(“set names GBK”)”实现。所以，用户在学习操作 MySQL 数据库的同时，还要与 PHP 函数联合应用。下面笔者为用户讲解 PHP 操作 MySQL 数据库的第一课。本实例通过 mysql_connect()函数连接 MySQL 数据库服务器，运行结果如图 2.10 所示。

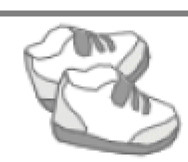
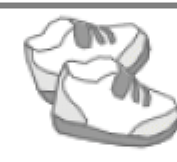


图 2.10　连接 MySQL 数据库服务器

实现过程

新建 index.php 文件，并创建 form 表单提交连接 MySQL 服务器的用户名、密码和服务器。获取 form 表单提交的数据，应用 mysql_connect()函数完成与 MySQL 服务器的连接。其代码如下：

```
<?php
if(!empty($_POST["sub"])){                                          //单击“确定”按钮
    //检测文本框是否有为空
  if (empty($_POST["na"]) || empty($_POST["nam"] ) || empty($_POST["name"]) ) {
      echo "<script>alert('文本框不能为空');</script>";              //输出提示
  }else{
```

```
            if(@mysql_connect("$_POST[na]","$_POST[nam]","$_POST[name]")){//连接 MySQL 服务器
                echo "<script>alert('连接 MySQL 服务器成功');</script>";            //输出提示
            }else{
                echo "<script>alert('错误了');</script>";
            }
        }
    }
?>
```

技术要点

mysql_connect()函数用于打开一个到 MySQL 服务器的连接。其语法如下：

```
resource mysql_connect ( [string server [, string username [, string password [, bool new_link [, int client_flags]]]]] );
```

参数说明：

☑ string server：表示连接数据库的 HOST。

☑ string username：表示连接 MySQL 数据库的用户名。

☑ string password：表示连接 MySQL 数据库的密码。

多学两招：

MySQL 数据库在安装时会提示用户设置 Host、用户名和密码。如果用户并未进行设置，MySQL 在默认安装的情况下，Host 为“localhost”，用户名为“root”，密码为“111”。

实例 026　选择指定的 MySQL 数据库

（实例位置：配套资源\SL\02\026　视频位置：配套资源\SP\02\026）

实例说明

通过函数 mysql_connect()可以与 MySQL 服务器建立连接，之后就可以操作数据库了。在“cmd”命令提示符下选择想要操作的数据库需要使用“use”语句，在 PHP 代码中需要使用函数 mysql_select_db()选择并连接数据库。本实例通过函数 mysql_select_db()实现与 MySQL 数据库的选择与连接，运行结果如图 2.11 所示。



图 2.11　选择指定的 MySQL 数据库



Note

实现过程

具体步骤如下：

（1）新建 index.php 文件。首先连接 MySQL 服务器，然后通过 mysql 函数库的函数读取输出 MySQL 数据库中存储的数据库名称，并且通过下拉列表框显示所有数据库名称。其代码如下：

```
<?php
$conn = mysql_connect("localhost","root","111");              //连接 MySQL 服务器
$rs = mysql_query("show databases");                          //查询操作
while($rst = mysql_fetch_array($rs)){                         //while 循环
  $a = 0;
  echo "<option value=".$rst[$a].">".$rst[$a]."</option><br>";   //输出数据库名称
  $a++;
}
?>
```

（2）当单击“确定”按钮时，将下拉列表框的选择信息以参数的形式传递给函数 mysql_select_db()，并输出提示信息。其代码如下：

```
<?php
if(! empty($_POST["sub"])){                                   //通过 post 方式传递参数
  if($_POST["select"] == "1"){                                //当下拉列表框值为 1 时
        echo "<script>alert('请选择数据库');</script>";          //输出提示信息
  }else{
        if(mysql_select_db("$_POST["select"]",$conn)){         //如果不为 1，连接数据库
              mysql_close();                                   //关闭连接
              echo "<script>alert('已选择指定数据库');</script>"; //输出提示
        }else{
              echo "<script>alert('出错误了');</script>";       //提示信息
        }
  }
}
?>
```

技术要点

mysql_select_db 函数用于选择 MySQL 数据库。其语法如下：

```
bool mysql_select_db ( string database_name [, resource link_identifier] );
```

参数说明：

☑　string database_name：表示选择 MySQL 数据库的名称。

☑　resource link_identifier：表示 MySQL 连接标识。

多学两招：

本实例是利用函数“mysql_select_db()”的返回值，判断是否连接到数据库，从而输出提示信息。还可以通过函数“mysql_select_db() or die()”来判断是否连接到数据库，如果连接到数据库将不会有任何信息提示，否则输出错误信息。





Note

实例 027　执行 SQL 语句

（实例位置：配套资源\SL\02\027　视频位置：配套资源\SP\02\027）

实例说明

用 PHP 代码操作 MySQL 数据库的主要目的是为了让程序控制数据库信息的增、删、改、查等操作。而函数 mysql_query()可以看作是部分 SQL 语句的载体。本实例通过 mysql_query()函数执行 SQL 语句，运行结果如图 2.12 所示。



图 2.12　通过 mysql_query()函数执行 SQL 语句

实现过程

新建 index.php 文件。首先，连接服务器和 db_database02 数据库。然后，获取 form 表单中提交的 SQL 语句，并且对 SQL 语句的类型进行判断，根据不同的类型应用 mysql_query()函数执行不同的操作。其关键代码如下：

```
<?php
if(!empty($_POST["sub"])){                                              //单击按钮
  $sql = $_POST[text"];                                                 //定义变量
  if($sql == ""){                                                       //如果文本框为空
       echo "<script>alert('您输入的文本框内容为空');</script>";          //JavaScript 提示
  }else{
       $conn = mysql_connect("localhost","root","111") or die("连接 MySQL 出现错误");//连接 MySQL
       mysql_select_db("db_database02",$conn) or die("false");          //连接数据库
       $sql1 = substr($sql,0,6);                                        //截取 SQL 语句前 6 个字符
       if(strtoupper($sql1) == "SELECT"){
            if(mysql_query($sql)){                                      //执行查询操作
                 echo "<script>alert('您在执行查询操作');</script>";
            }else{
                 echo "<script>alert('SQL 语句出现错误');</script>";
            }
       }else{
            if(strtoupper($sql1) == "INSERT"){
                 if(mysql_query($sql)){
```





```
                echo "<script>alert('您在执行插入操作');</script>";
            }else{
                echo "<script>alert('SQL 语句出现错误');</script>";
            }
        }else{
            if(strtoupper($sql1) == "UPDATE"){
                if(mysql_query($sql)){
                    echo "<script>alert('您在更新查询操作');</script>";
                }else{
                    echo "<script>alert('SQL 语句出现错误');</script>";
                }
            }
        }
    }
  }
}
?>
```

指点迷津：

在对 SQL 语句的类型进行判断时，首先应用 substr()函数对 SQL 语句进行截取，截取其前 6 个字节，然后将截取的内容转换为小写，并通过这 6 个字符串对 SQL 语句的类型进行判断。

技术要点

mysql_query()函数用于向与指定的连接标识符关联的服务器中当前活动的数据库发送一条 MySQL 查询。其语法如下：

```
resource mysql_query ( string query [, resource link_identifier] );
```

参数说明：

☑ resource link_identifier：连接标识。

☑ string query：SQL 语句。

多学两招：

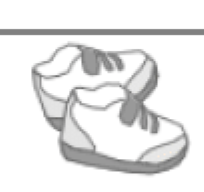
mysql_query()用于向与指定的连接标识符关联的服务器中当前活动的数据库发送一条查询。如果没有指定 link_identifier，则使用上一个打开的连接。如果没有打开的连接，本函数会尝试无参数调用 mysql_connect() 函数来建立一个连接并使用，其查询结果会被缓存。

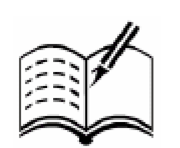
实例 028　文本文件分页读取

（实例位置：配套资源\SL\02\028　视频位置：配套资源\SP\02\028）

实例说明

在遍历文件中的内容时，由于文件内容很大，所以最理想的方法就是分页读取文本文







件中的内容。本实例中将介绍如何通过自定义函数完成文本文件中数据的分页读取，运行结果如图 2.13 所示。



图 2.13　文本文件的分页读取

实现过程

具体步骤如下：

（1）创建 function.php 文件，并通过自定义函数 msubstr()完成对文本文件的截取操作。

（2）创建 index.php 文件。首先通过文件系统函数 file_get_contents()读取整个文件的内容，然后调用自定义函数和字符串函数完成对文件的截取操作，实现截取后内容的分页输出。其关键代码如下：

```
<div id="synopsis">
<!--创建 div 标签，用于获取 js 文件中返回的分页结果-->
    <table width="545" border="0" cellspacing="0" cellpadding="0">
        <tr>
            <td><?php
                     include("function.php");
//读取超长文本中的数据，实现超长文本中数据的分页显示
  if($_GET['page']){
   $counter=file_get_contents("data.txt");
      $length=strlen($counter);
      $page_count=ceil($length/850);
      $c=msubstr($counter,0,($_GET['page']-1)*850);
      $c1=msubstr($counter,0,$_GET['page']*850);
       echo substr($c1,strlen($c),strlen($c1)-strlen($c));
  }
?>
            </td>
        </tr>
        <tr>
            <td><table width="535" border="0" align="center" cellpadding="0" cellspacing="0">
                <tr>
                <!--设置超长文本分页显示的超连接-->
                     <td width="37%" align="center" valign="middle" bgcolor="#FFFFFF">内
容 <?php echo $page_count;?> 页  当前 第 <?php echo $_GET['page'];?> 页</td>
```





```
                        <td  width="63%"  height="28"  align="center"  valign="middle"  bgcolor=
"#FFFFFF"><?php
                   if($_GET['page']!=1){
     ?>
                   <!--调用 no_refurbish_pagination 函数，实现无刷新的分页输出-->
                        <a  href="#"  onClick='return  no_refurbish_pagination("index_ok.php?
page=1")'>首页</a>  <a href="#" onClick='return no_refurbish_pagination("index_ok.php?page=<?php
echo $_GET['page']-1;?>")'>上一页</a>
                        <?php  }
                         if($_GET['page']<$page_count){
     ?>
                        <a href="#" onClick="return no_refurbish_pagination('index_ok.php?page=
<?php echo $_GET['page']+1;?>')">下一页</a> <a href="#" onClick='return no_refurbish_pagination("index_
ok.php?page=<?php echo $page_count;?>")'>尾页</a>
                        <?php
                   }
                       ?>
                        </td>
                     </tr>
                   </table></td>
          </tr>
        </table>
     </div>
```

（3）创建 index_ok.php 文件，实现数据的无刷新分页。index_ok.php 中的内容与 index.php 中 div 标签 synopsis 的内容相同。

（4）创建 js 脚本文件 discuss_js.js，通过 Ajax 实现无刷新的操作。

技术要点

完成超长文本的分页输出需要三方面的技术：第一方面，自定义函数。通过自定义函数读取文本文件，可以避免中文字符串出现乱码；第二方面，字符串函数。需要通过 strlen()函数计算字符串的长度，通过 substr()函数对字符串进行截取；第三方面，文件系统函数。通过 file_get_contents()函数读取文本文件中的数据。

自定义函数 msubstr()的结构如下：

```
     //定义一个用于截取一段字符串的函数 msubstr()
     function msubstr($str,$start,$len){//$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的
是长度
     $strlen=$start+$len;//用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
     $tmpstr= "";                      //设置变量值为空
     for($i=0;$i<$strlen;$i++){        //通过 for 循环语句，循环读取字符串
       if(ord(substr($str,$i,1))>0xa0){//如果字符串中首个字节的 ASCII 序数值大于 0xa0，则表示为汉字
       $tmpstr.=substr($str,$i,2);     //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
           $i++;                       //变量自加 1
       }else{                          //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
               $tmpstr.=substr($str,$i,1);
     }
```



```
    }
    return $tmpstr;                    //输出字符串
}
```



Note

多学两招：

在本实例中，不但实现了文本文件内容的分页输出，而且是无刷新分页输出。具体操作通过 js 脚本文件 discuss_js.js 和 index_ok.php 文件完成。

实例 029　查询关键字描红

（实例位置：配套资源\SL\02\029　视频位置：配套资源\SP\02\029）

实例说明

关键字描红技术广泛应用于站内搜索或者高级搜索中，是一项很常用的技术。本实例通过函数 str_replace()实现查询关键字描红，运行结果如图 2.14 所示。

被搜索的文本：PHP作为全球最普及、应用最广泛的互联网开发语言之一，从1994年诞生至今已被2000多万个动态网站采用，全球知名互联网公司Google、Yahoo、eBay和中国知名网站新浪、百度、阿里巴巴等均采用PHP技术！
关键字：PHP 搜索
PHP作为全球最普及、应用最广泛的互联网开发语言之一，从1994年诞生至今已被2000多万个动态网站采用，全球知名互联网公司Google、Yahoo、eBay和中国知名网站新浪、百度、阿里巴巴等均采用**PHP**技术！

图 2.14　查询关键字描红

实现过程

创建 PHP 脚本文件，当单击“搜索”按钮时，首先通过 post 方法将接收到的数据进行描红处理，然后应用 str_replace()函数对指定字符串信息的指定内容进行替换。其代码如下：

```
<?php
if(!empty($_POST["sub"])){
  $a = "<b style='color:red;font-size:18px;'>".$_POST["text"]."</b>";  //对提交的关键字进行描红
  echo str_replace("$_POST["text"]",$a,$str);                           //完成字符串的替换操作
  }
?>
```

技术要点

str_replace ()函数用于执行字符串替换操作。其语法如下：

```
mixed str_replace ( mixed search, mixed replace, mixed subject [, int &count] )
```

参数说明：

☑ search：必选参数，指定需要查找的字符串。

☑ replace：必选参数，指定替换的值。

☑ subject：必选参数，指定的查找范围。

☑ count：可选参数，获取执行替换的数量。

多学两招：

不但可以通过 str_replace()函数实现关键字描红操作，还可以通过 str_ireplace()函数实现，并且该函数还不区分大小写。





实例 030　分页显示图书信息

（实例位置：配套资源\SL\02\030　视频位置：配套资源\SP\02\030）

实例说明

select 语句可以查询数据库信息，但是如果数据库信息非常多，在一个页面中显示所有数据，不仅会使页面显得臃肿、不美观，而且会给用户造成网站不够智能的假象。本实例灵活运用 select 语句实现对图书信息的分页处理，运行结果如图 2.15 所示。

分页显示图书信息

查询图书信息

首页 上一页 下一页 尾页

ID	书名	价格	日期
6	《C#开发实战宝典》	75元	2010-07-16
7	《C++开发实战宝典》	75元	2010-07-16
8	《PHP范例宝典》	75元	2010-07-16
9	《VB范例宝典》	75元	2010-07-16
10	《JAVA范例宝典》	75元	2010-07-16

图 2.15　分页显示图书信息

实现过程

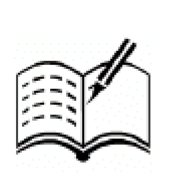
具体步骤如下：

（1）创建 in.php 文件。通过面向对象的知识编写数据库连接类，并将数据库连接操作封装入类。

（2）创建 index.php 文件。首先，包含数据库连接类并实例化对象。其次，定义查询语句统计查询结果集中的记录数，并定义分页变量。然后，定义分页查询语句，每页显示 5 条记录。最后，创建分页超链接，完成图书信息的分页显示。其核心代码如下：

```
<?php
include("in.php");                                         //包含数据库连接类
new mysql("localhost","root","111","db_database02");       //实例化对象
if(empty($_GET["id"] ) || $_GET["id"] == 1){               //如果地址栏参数等于 1
  $num = 5;                                                //定义变量
  $rse = mysql_query("select * from tb_demo026");          //执行查询操作
  $nu = mysql_num_rows($rse);                              //返回数据库信息条数
  if(empty($_GET["page"] )){                               //设置地址栏参数 page
        $page = 1;
  }else{
        $page = $_GET["page"];
  }
        $sql = "select * from tb_demo026 limit ".(($page - 1)*$num).",".$num;//分页 sql 语句
        $rs = mysql_query($sql);                           //返回结果集
        $nur = ceil($nu/$num);                             //取整函数
        echo "<a href='index.php?id=1&page=1'>首页</a>";    //设置主页地址栏参数
        if($page >= 2){                       //如果地址栏参数 page 值大于等于 2
```






```
?>
        <a href="index.php?id=1&page=<?php echo $page-1;?>">上一页</a>      //上一页
<?php
    }
    if($page < $nur){                                    //如果 page 小于总页数
?>
        <a href="index.php?id=1&page=<?php echo $page+1;?>">下一页</a>      //下一页
<?php
    }
?>
        <a href="index.php?id=1&page=<?php echo $nur;?>">尾页</a>           //尾页
<?php
    echo "<table width='580px'><tr><td width='100px' bgcolor='#FFFFFF'>ID</td><td width=
'200px'>书名</td><td width='180px' bgcolor='#FFFFFF'>价格</td><td width='100px'>日期</td></tr>";
    while($rst = mysql_fetch_row($rs)){
        echo "<tr><td bgcolor='#FFFFFF'>".$rst[0]."</td><td>《".$rst[1]."》</td><td bgcolor=
'#FFFFFF'>".$rst[2]."</td><td>".$rst[3]."</td></tr>";
    }
    echo "</table>";
  }
?>
```

脚下留神：

其中 if 判断语句内 empty($_GET["id"]) || $_GET["id"] == 1 的作用是判断 URL 地址中是否存在 id 或 id 值为“1”。如果值判断获取的 id 数为“1”，当运行程序时，可能会因无法获取 id 值而导致不必要的错误，所以这里的判断条件不可省略。

技术要点

本实例应用 select 查询语句及其子句 limit 实现图书信息的分页显示，通过 limit 关键字控制当前页面显示的记录数和开始位置。其分页查询图书信息的 SQL 语句代码如下：

```
$sql = "select * from tb_demo026 limit ".(($page - 1)*$num).",".$num;
```

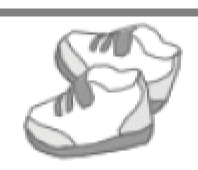
其中(($page - 1)*$num)计算的是当前页的开始位置，而$num 是当前页显示的记录数。

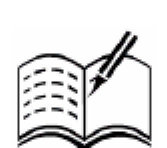
指点迷津：

select 语句中的 limit 子句功能很强大，通过它可以控制查询语句的开始位置，以及查询的数量。但是，这个 limit 子句是 MySQL 数据库特有的，在 SQL Server 或者 Access 数据库中是不支持的。

多学两招：

如果要提高网站的安全性，可以防止非法用户进入网站，即在用户进入网站前进行注册，只有注册成功的用户才可以进入网站。下面将通过几个实例介绍如何设计用户注册。实例 031～033 向读者展示了如何实现用户注册的一般过程。





实例 031 一般用户注册

（实例位置：配套资源\SL\02\031 视频位置：配套资源\SP\02\031）

实例说明

在开发网络程序时，很多时候都会应用到用户注册模块。而用户注册模块有简单和复杂之分，本实例就是一个相对比较简单的用户注册程序。用户注册主要就是为了保存用户的一些基本信息，包括用户名、真实姓名、密码等。运行本实例，在用户注册页面中添加用户的基本信息，填写完成后，单击“提交”按钮，将用户注册信息保存到数据库中。本实例的运行结果如图 2.16 所示。

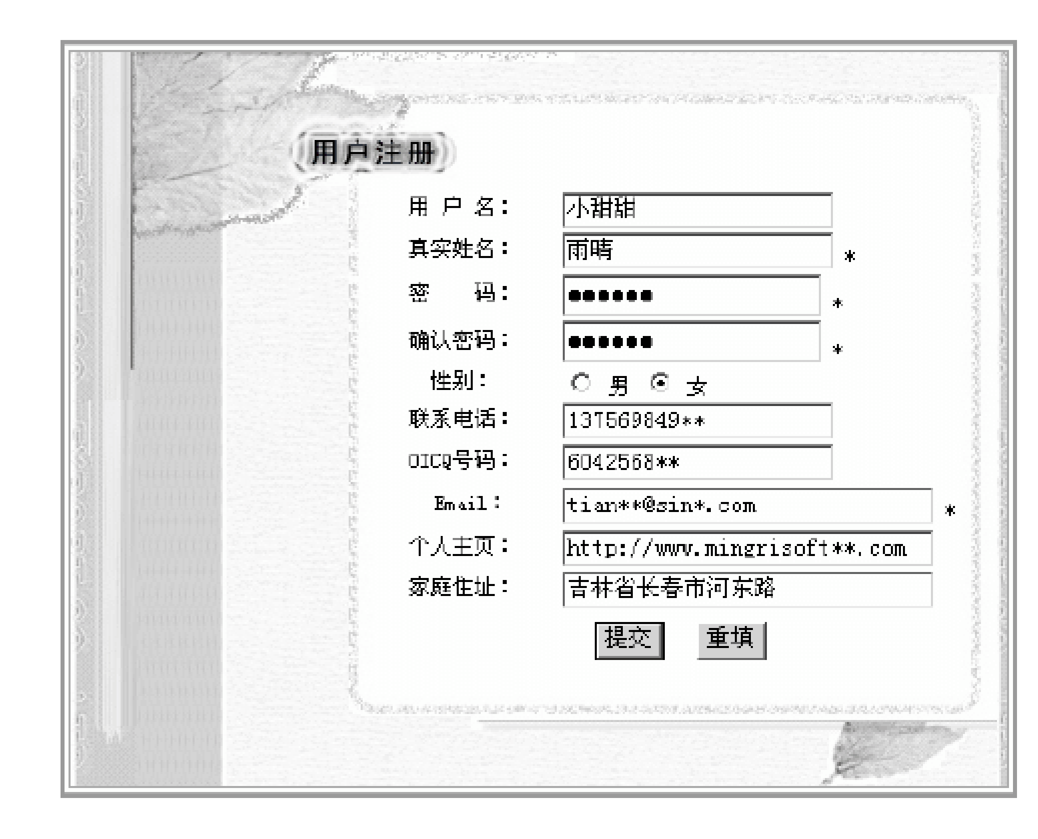


图 2.16 一般用户注册

实现过程

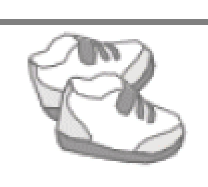
具体步骤如下：

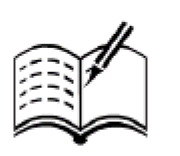
（1）创建与数据库的连接，代码如下：

```
<?php
  $link=mysql_connect("localhost","root","111");
  mysql_select_db("db_database02",$link);
  mysql_query("set names gb2312");
?>
```

（2）添加 form 表单、文本框、单选按钮、按钮，并设置相关属性值。其关键代码如下：

```
<form action="register_deal.php" method="post" name="myform" >
  <input name="username" type="text" id="username" size="20">
  <input name="truename" type="text" id="truename" size="20">
  <input name="pwd1" type="password" id="pwd1" size="20">
  <input name="pwd2" type="password" id="pwd2" size="20" onBlur="javascript:if(this.value!=this.
form.pwd1.value){ alert('您两次输入的密码不一致！');}">
  <input name="sex" type="radio" value="男" checked>
      <input type="radio" name="sex" value="女">
      <input name="tel" type="text" id="tel">
      <input name="oicq" type="text" id="oicq">
      <input name="email" type="text" id="email" size="28">
  <input name="homepage" type="text" id="homepage" size="28">
      <input name="address" type="text" id="address" size="28">
  <input name="submit" type="submit" class="btn_grey" value="提交" onClick="return check()">
  <input name="submit2" type="reset" class="btn_grey" value="重填">
</form>
```





（3）当用户按要求填写完注册信息后，单击“提交”按钮，程序会将用户录入的注册信息提交到数据处理页（register_deal.php）进行数据处理。首先，载入数据库连接文件，然后，将从表单中提交的数据分别存储到变量中，使用 insert 语句和 mysql_query()函数将这些数据作为新记录插入到数据表中。其关键代码如下：

```
<?php
include "conn/conn.php";                    //载入数据库连接文件
if($_POST["submit"]!=""){                   //判断提交按钮
  $username=$_POST["username"];
  $truename=$_POST["truename"];
  $pwd=$_POST["pwd1"];
  $sex=$_POST["sex"];
  $email=$_POST["email"];
  $tel=$_POST["tel"];
  $homepage=$_POST["homepage"];
  $oicq=$_POST["oicq"];
  $address=$_POST["address"];
  $sql="insert into tb_user (username,pwd,truename,sex,email,tel,homepage,oicq,address) Values
('$username','$pwd','$truename','$sex','$email','$tel','$homepage','$oicq','$address')";
  $result=mysql_query($sql);          //执行添加操作
  if($result){
      echo "<script>alert('用户注册成功！');window.location='index.php';</script>";
  }else{
      echo "<script>alert('用户注册失败！');window.location='index.php';</script>";
  }
}
?>
```

技术要点

本实例主要应用表单提交用户注册信息，应用 insert into 语句和 mysql_query()函数完成注册信息向数据表中添加的操作。

mysql_query()函数用来根据连接标识符向数据库服务器的当前数据库发送查询。其语法格式如下：

```
int mysql_query(string query ,int [link_identifier]);
```

其中，query 是查询字符串；link_identifier 是数据库连接标识符。

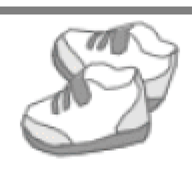
mysql_query()在执行成功时返回一个结果标识符，失败时返回 false。

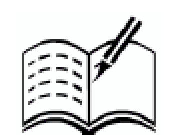
实例 032　带检测用户名的用户注册

（实例位置：配套资源\SL\02\032　视频位置：配套资源\SP\02\032）

实例说明

在开发网络程序时，用户注册的形式多种多样。除了上面讲解的一般用户注册外，还





Note

可以对相同的用户名进行检测，其目的在于检测用户名是否已经被占用。考虑到用户的权限问题，一般的管理网站不允许同时存在两个或多个相同的用户名，该方法提高了网站的安全性。运行本实例，在“用户名”的文本框中输入用户名，通过单击“检测用户”超链接对用户输入的新用户名进行检测，从而判断该用户名是否已存在。如果已存在，则提示用户重新输入用户名；否则，用户可以进行其他相关信息的输入。其运行结果如图 2.17 所示。

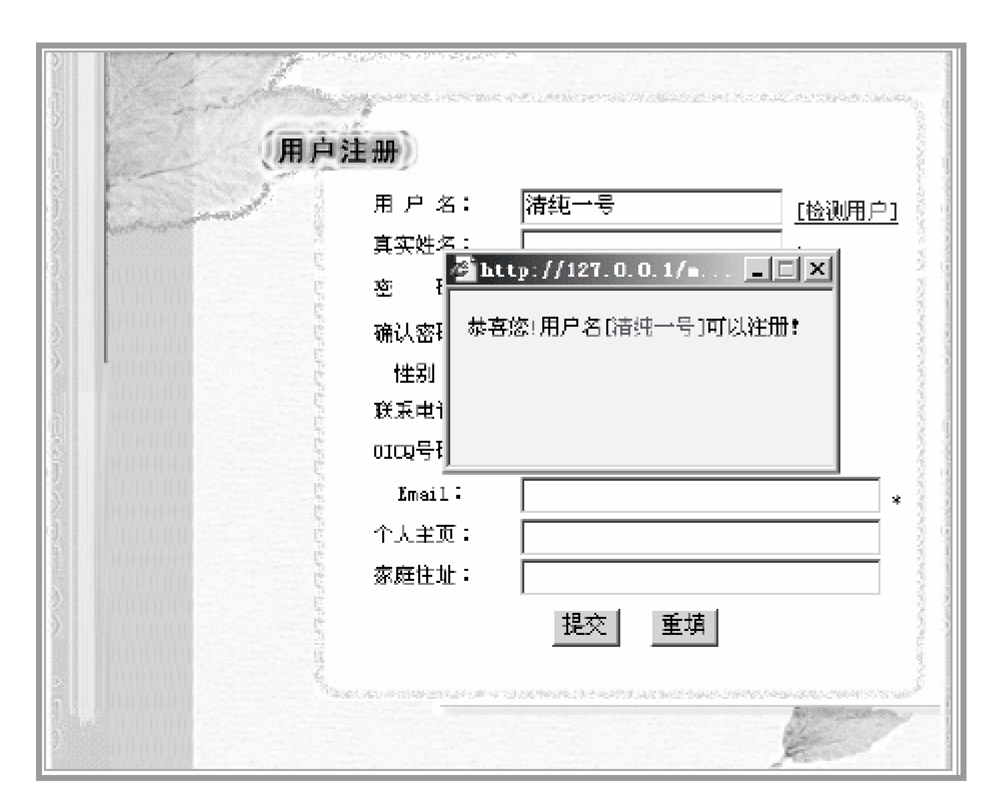


图 2.17　带检测用户名的用户注册

实现过程

具体步骤如下：

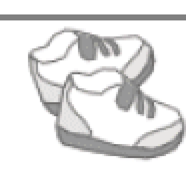
（1）在注册用户时，一般不允许同时存在相同的用户名，其目的是为了便于网站管理员区分不同的注册用户。单击“检测用户”超链接，系统就会调用 JavaScript 脚本的 openwin()函数，将用户提交的用户名传递到 submit_checkuser.php 文件中，并完成对这个用户名的检测操作。其代码如下：

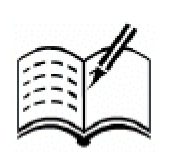
```
<a href="#" onclick="javascript:openwin(myform.username.value)">[检测用户]</a>
<script language="javascript">
function openwin(x){
  if (x==""){                              //判断用户名是否为空
        alert("请输入用户名!"); myform.username.focus();return false;
  }else{                                   //执行用户名的检测操作
        window.open("submit_checkuser.php?x="+x,"newframe","width=220,height=60");
  }
}
</script>
```

（2）创建 submit_checkuser.php 文件，通过$_GET 方法获取超链接传递的用户名，在 submit_checkuser.php 文件中完成对用户名的检测操作。其关键代码如下：

```
<?php
header("Content-Type:text/html; charset=gb2312");
include "conn/conn.php";
$username=$_GET['x'];
$sql=mysql_query("select * from tb_consumer where username = '$username'");
$result=mysql_fetch_array($sql);
if($result!=false){
  echo ("[<font color=red>".$username."</font>]已被注册！");
}else{
  echo ("恭喜您!用户名[<font color=green>".$username."</font>]可以注册！");
}
?>
```

（3）创建 register_deal.php 文件，获取表单提交的数据，完成用户的注册操作。其关





键代码如下：

```
<?php
header("Content-Type:text/html; charset=gb2312");
include "conn/conn.php";            //连接数据库
if($_POST["submit"]!=""){
  $username=$_POST['username'];
  $truename=$_POST['truename'];
  $pwd=$_POST['pwd1'];
  $sex=$_POST['sex'];
  $email=$_POST['email'];
  $tel=$_POST['tel'];
  $homepage=$_POST['homepage'];
  $oicq=$_POST['oicq'];
  $address=$_POST['address'];
  $sql="insert  into  tb_consumer  (username,pwd,truename,sex,email,tel,homepage,oicq,address)
Values('$username','$pwd','$truename','$sex','$email','$tel','$homepage','$oicq','$address')";
  $result=mysql_query($sql);  //执行添加操作
  if($result){
       echo "<script>alert('用户注册成功！');window.location='index.php';</script>";
  }else{
       echo "<script>alert('用户注册失败！');window.location='index.php';</script>";
  }
}
?>
```

技术要点

本实例的关键是通过超链接中的 onclick 调用 JavaScript 脚本中的 open 方法打开一个窗口，提交用户名，在打开的窗口文件中完成对用户名的检测操作，并且返回检测结果。

open 方法可以打开一个新的窗口，并在窗口中装载指定 URL 地址的网页。其语法如下：

```
WindowVar=window.open(url,windowname[,location]);
```

WindowVar：当前打开窗口的句柄。如果 open()方法成功，则 WindowVar 的值为一个 window 对象的句柄；否则，WindowVar 的值是一个空值。

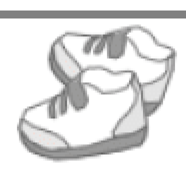
url：目标窗口的 URL。如果 URL 是一个空字符串，则浏览器将打开一个空白窗口，允许用 write()方法创建动态 HTML。

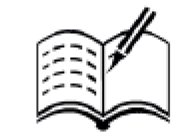
windowname：window 对象的名称。该名称可以作为属性值在<a>和<form>标记的 target 属性中出现。如果指定的名称是一个已经存在的窗口名称，则返回对该窗口的引用，而不会再打开一个新的窗口。

location：对对话框属性进行设置的可选参数如表 2.1 所示。

表 2.1　location 属性的可选参数值

属　　性	描　　述
top	窗口顶部离开屏幕顶部的像素数
left	窗口左端离开屏幕左端的像素数
width	对话框的宽度





续表

属　　性	描　　述
height	对话框的高度
scrollbars	是否显示滚动条
resizable	设定对话框大小是否固定
toolbar	浏览器工具条，包括后退及前进按钮等
menubar	菜单条，一般包括有文件、编辑及其他一些条目
location	定位区，也叫地址栏，是可以输入 URL 的浏览器文本区
direction	更新信息的按钮

实例 033　分步用户注册

（实例位置：配套资源\SL\02\033　视频位置：配套资源\SP\02\033）

实例说明

除了上面讲到的两种用户注册方法外，用户注册还可以实现分步注册的功能。分步注册可以让用户更清楚地了解用户具有哪些权限及哪些特殊的功能。运行本实例，单击“同意以上条款”按钮，表示在注册成为用户时，已确认该服务条款，此时将进入到用户注册页面。如果单击“不同意”按钮，将不能进行用户注册。进入用户注册页面时，按要求填写相关的用户信息。填写完成后，再单击“提交”按钮，完成用户信息的注册。本实例的运行结果如图 2.18 和图 2.19 所示。

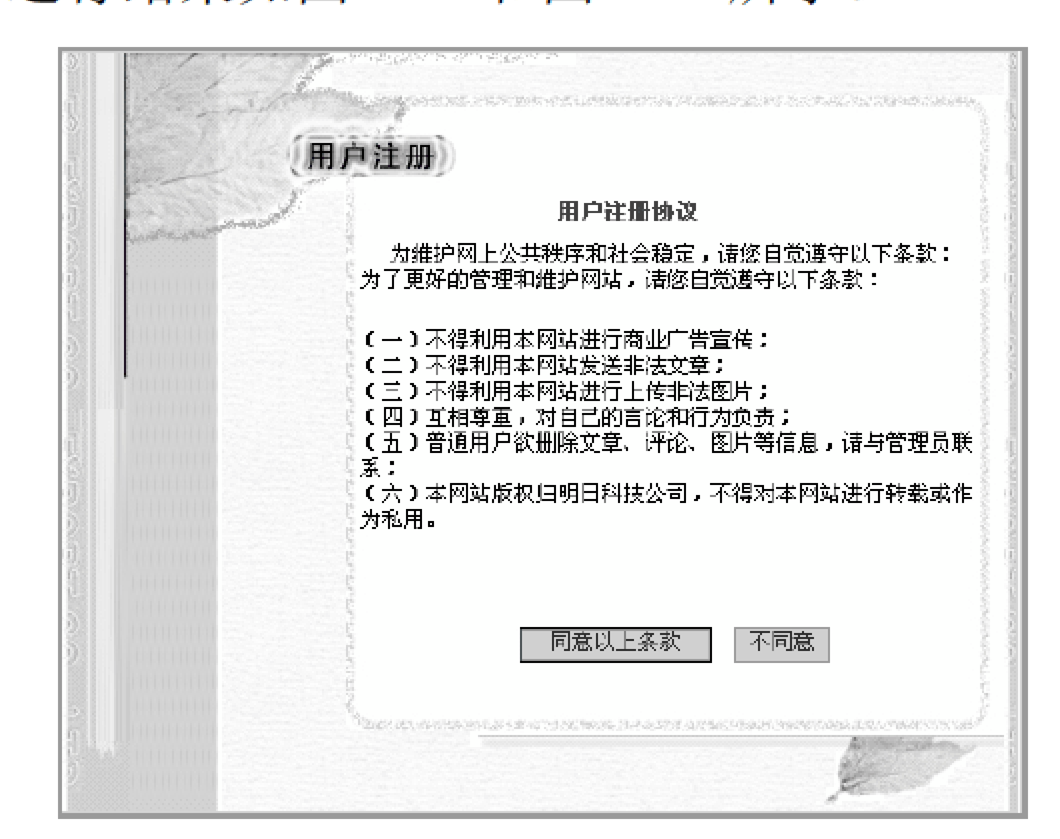


图 2.18　用户注册协议

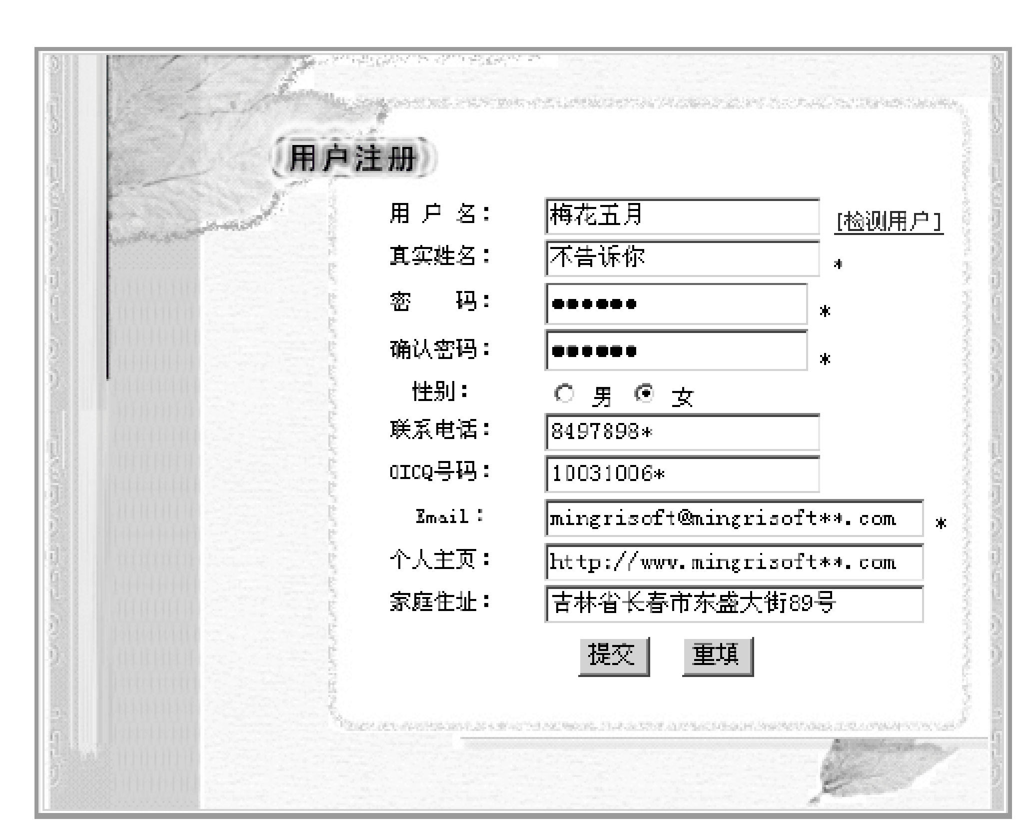


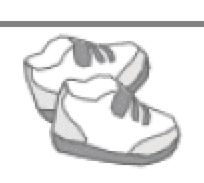
图 2.19　分步用户注册

实现过程

具体步骤如下：

（1）首先确认是否接受服务条款，单击“同意以上条款”按钮将进入用户注册页面；单击“不同意”按钮，则不能进入到注册页面。在 index.php 文件中，输出注册服务条款，以及处理按钮。其关键代码如下：

```
    <input id="Button4" style="font-size: 9pt" type=submit value="同意以上条款 " name="Button1"
onClick="window.location.href='register.php'">
```



```
<input id="Button2" style="font-size: 9pt" type="submit" value=" 不 同 意 " name="Button2" onClick="window.location.href='index.php'">
```

（2）在 register.php 文件中创建 JavaScript 脚本，并通过自定义方法验证用户输入的信息是否合法。其关键代码如下：

```
<script  language="javascript">
//判断用户的输入是否合法
function check(){
  if (myform.truename.value==""){
       alert("请输入真实姓名！");myform.truename.focus();return false;
  }
//省略部分代码
  var i=myform.email.value.indexOf("@");
  var j=myform.email.value.indexOf(".");
  if((i<0)||(i-j>0)||(j<0)){
       alert("您输入的 Email 地址不正确，请重新输入！");myform.email.value="";myform.email.focus();return false;
  }
  myform.submit();
}
</script>
```

（3）在 register.php 文件中创建“检测用户”超链接，编写 JavaScript 脚本的 openwin()方法，调用 submit_checkuser.php 文件对用户名进行检测。

（4）创建 register_deal.php 文件，获取表单提交的用户注册信息，并应用 insert to 语句和 mysql_query()将用户注册信息添加到指定的数据表中。

技术要点

本实例在完成用户的注册功能时，增加了一个服务器条款的阅读过程。只有在同意服务条款中的约定之后，才能注册为本站的会员。

另外，在本实例中所有调用 JavaScript 脚本的操作，都是通过鼠标的单击事件（onclick）来完成的。

单击事件（onclick）是指在鼠标单击时被触发的事件。单击是指鼠标停留在对象上，按下鼠标键，在没有移动鼠标的同时放开鼠标键的这一完整过程。

单击事件一般应用于 Button 对象、Checkbox 对象、Image 对象、Link 对象、Radio 对象、Reset 对象和 Submit 对象，Button 对象一般只会用到 onclick 事件处理程序，因为该对象不能从用户那里得到任何信息，如果没有 onclick 事件处理程序，Button 对象将不会有任何作用。

指点迷津：

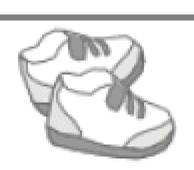
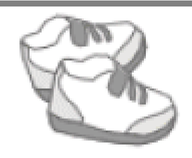
在使用对象的单击事件时，如果在对象上按下鼠标键，然后移动鼠标到对象外再松开鼠标，那么单击事件就会无效。单击事件必须在对象上按下并松开后，才会执行单击事件的处理程序。



PHP 流程控制语句

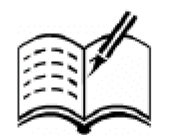
本章读者可以学到如下实例：

- ⏭ 实例 034　员工生日提醒
- ⏭ 实例 035　SWITCH 网页框架
- ⏭ 实例 036　员工信息的批量删除
- ⏭ 实例 037　do…while 语句循环读取数据库中数据
- ⏭ 实例 038　生成随机验证码
- ⏭ 实例 039　图形计数器
- ⏭ 实例 040　包含数据库连接文件
- ⏭ 实例 041　健康生活提醒
- ⏭ 实例 042　if 语句判断美女征婚条件
- ⏭ 实例 043　网页版九九乘法表
- ⏭ 实例 044　读取购物车中的数据
- ⏭ 实例 045　多图片上传
- ⏭ 实例 046　控制页面中表情图的输出
- ⏭ 实例 047　控制页面中数据的输出数量
- ⏭ 实例 048　考试成绩评定标准

实例 034　员工生日提醒

（实例位置：配套资源\SL\03\034　视频位置：配套资源\SP\03\034）



Note

实例说明

在 PHP 应用中，类似生日提醒等功能的程序随处可见，主要是因为这样的程序有一定的定时效果。本实例通过 if 语句和 foreach 循环语句实现一个员工生日提醒的小程序，运行结果如图 3.1 所示。

员工生日提醒程序

当前日期：07月15日

今天是小杨的生日

图 3.1　员工生日提醒

实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，根据数组的特点将键值与对应值存储在数组之中。然后，利用 foreach 语句遍历数组，将取得的 value 值与当前日期进行比较。最后，将符合条件的值进行输出。其代码如下：

```
<?php
    $a = array("小杨"=>"07 月 15 日","小王"=>"07 月 28 日","小张"=>"07 月 29 日","小李"=>"07 月 30 日","小赵"=>"07 月 31 日","小刘"=>"06 月 1 日");
        foreach($a as $key => $value){              //定义 foreach 语句
            if(date("m 月 d 日") == $value){   //判断与当前日期是否相同
                echo "  今天是".$key."的生日";
            }
        }
?>
```

（2）运行本实例，结果如图 3.1 所示。

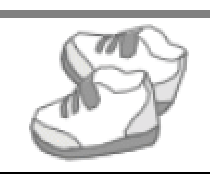
技术要点

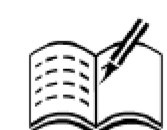
本实例的关键点是利用 foreach 语句循环读取员工信息；利用 if 选择语句，将文本框中的日期与数组中员工日期进行比较。

（1）foreach 语句：foreach 仅能用于数组，当试图将其用于其他数据类型或者一个未初始化的变量时，会产生错误。

```
foreach (array_expression as $value) statement;
```

参数 array 表示要遍历的数组，expression 表示键值，$value 表示键值的对应值，statement 表示语句块。





多学两招：

foreach 所操作的数据是指定数组的一个副本，而不是该数组本身。因此，数组指针不会被 each()结构改变，对返回的数组单元的修改也不会影响原数组。不过原数组的内部指针在处理数组的过程中的确向前移动了。假定 foreach 循环运行到结束，原数组的内部指针将指向数组的结尾。foreach 语句不支持用“@”来禁止错误信息。

（2）if 语句：if 语句对某段程序的执行附加一个条件，如果条件成立，就执行这段程序；否则，就跳过这段程序。

```
if(expr) statement;
```

参数 expr 表示判定条件，statement 表示语句块。

（3）array()函数：返回根据参数建立的数组。参数可以用 => 运算符给出索引。

```
array array ( [mixed ...]);
```

实例 035　SWITCH 网页框架

（实例位置：配套资源\SL\03\035　视频位置：配套资源\SP\03\035）

实例说明

类似留言板等功能较单一的网站，可以通过 switch 语句制作网页框架，将所有内容包含到主页当中。本实例通过 switch()语句和 include()语句来演示网页框架的制作，运行结果如图 3.2 所示。

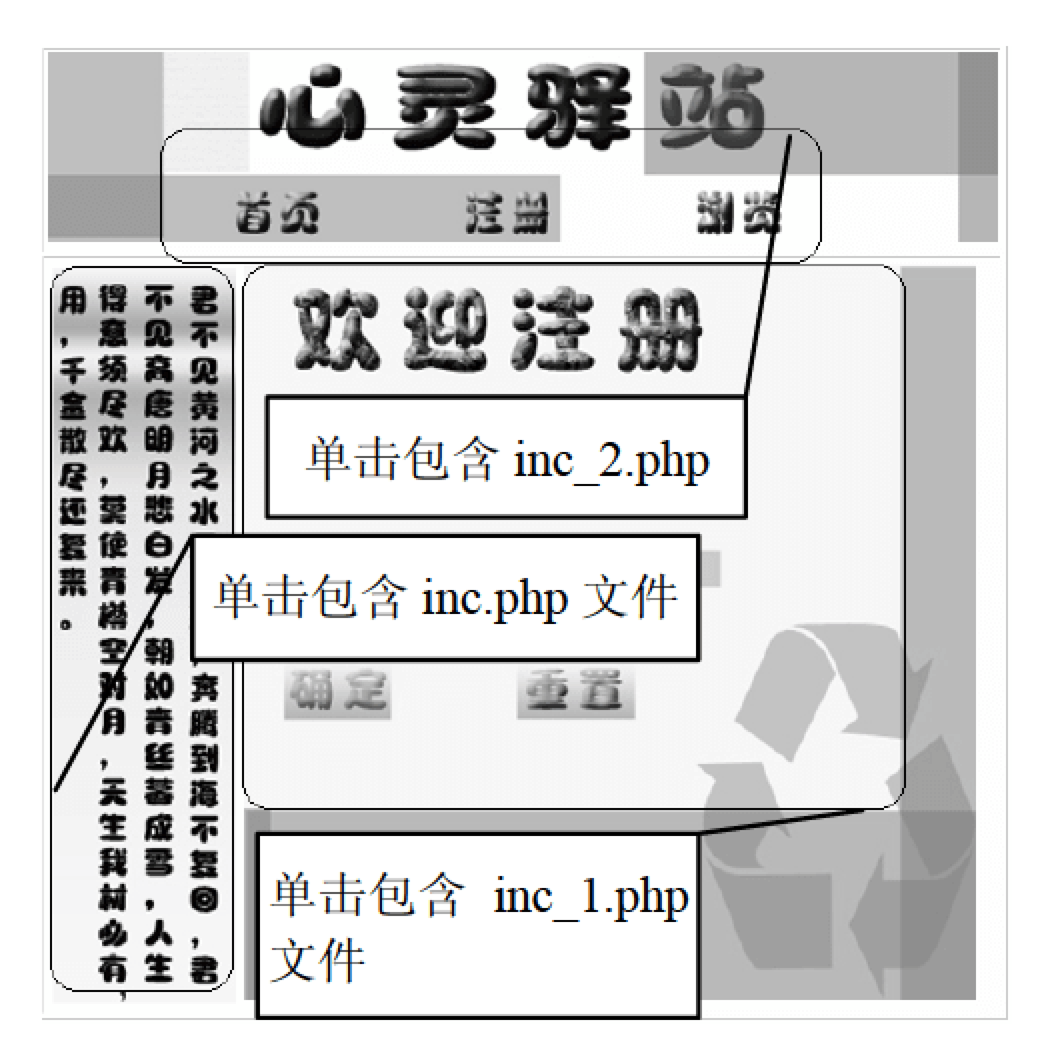


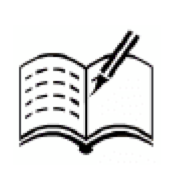
图 3.2　网页框架的制作

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，利用 switch 语句获取从地址栏传递的数据，并且将





数据与 switch 语句中的 case 做比较，当地址栏传递的数据与 case 中的数据相等时，包含指定文件。然后，利用 break 语句跳出 switch 循环。其代码如下：

```
<?php
    switch($_GET['link']){                         //通过地址栏接收参数
        case "主页";                                //如果值问主页
        include('inc.php');                        //包含“inc.php”文件
        break;                                     //结束执行 switch 语句
        case "注册";                                //如果值为注册
        include('inc_1.php');
        break;
        case "浏览";                                //如果值为浏览
        include('inc_2.php');
        break;
        default:                                   //默认情况下，包含文件“inc.php”
        include('inc.php');
    }
?>
```

（2）将该文件存储于\MR\03\035\文件夹下，并命名为 index.php。运行结果如图 3.2 所示。

脚下留神：

switch 语句在执行时，即使遇到符合要求的 case 语句段，也会继续往下执行，直到 switch 语句结束。为了避免这种浪费时间和资源的行为，一定要在每个 case 的语句段后添加 break 跳转语句跳出当前循环。

技术要点

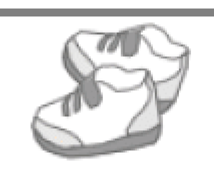
switch 语句和 break 语句的灵活运用是本实例的关键。

（1）switch 语句和具有同样表达式的一系列的 if 语句相似。很多场合下，需要把同一个变量（或表达式）与很多不同的值比较，并根据它等于哪个值来执行不同的代码。

switch 语句的语法格式如下：

```
switch(variable){
  case value1:
        statement1;
        break;
  case value2:
  …
  default:
        default statement n;
}
```

switch 语句根据 variable 的值，依次与 case 中的 value 值相比较。如果不相等，则继续查找下一个 case；如果相等，就执行对应的语句，直到 switch 语句结束或遇到 break 为止。一般 switch 语句最终都有一个默认值 default，如果在前面的 case 中没有找到相符的条件，则输出默认语句，这与 else 语句类似。





（2）用 break 结束当前 for、foreach、while、do…while 或者 switch 结构的执行。

实例 036　员工信息的批量删除

（实例位置：配套资源\SL\03\036　视频位置：配套资源\SP\03\036）

实例说明

在操作数据库的过程中，可能会出现很多无用的冗余信息。想要删除它们，用户可以使用 while 循环语句循环删除。本实例通过 while 循环语句实现员工信息的批量删除，运行结果如图 3.3 所示。

删除 3 条记录 删除
姓名为阳光的员工已被删除
姓名为郭靖的员工已被删除
姓名为郭靖的员工已被删除

图 3.3　员工信息的批量删除

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，连接 MySQL 数据库服务器，选择 MySQL 中名称为 db_while 的数据库，并设置页面的编码格式。然后，定义循环变量$a 和$b，再将第一层 while 循环中的循环条件设置为$b<$a。最后，再次利用 while 语句循环输出数据表中的数据，并执行数据的删除操作。其代码如下：

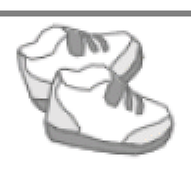
```
<?php
if(isset($_POST['sub'])){                                          //通过 POST 方式获取参数
  $conn = mysql_connect("localhost","root","111");                 //连接 mysql 数据库
  mysql_select_db("db_database03",$conn);                          //连接数据库
  mysql_query("SET NAMES utf8");                                   //定义编码格式
  $a = $_POST['te'];                                               //接收文本框参数
  $b = 0;                                                          //定义变量
  while($b < $a){                                                  //while 循环
        $rs = mysql_query("select * from tb_while");               //执行查询操作
        while($rst = mysql_fetch_array($rs)){                      //将查询结果保存在数组中
              $sql = "delete from tb_while where id = $b";         //sql 语句
              mysql_query($sql);                                   //执行删除操作
              echo "姓名为".$rst['name']."的员工已被删除<br>";      //输出被删除的员工姓名
              $b++;
        }
  }
}
?>
```

（2）将该文件存储于\MR\03\036 文件夹下，并命名为 index.php。运行结果如图 3.3 所示。

技术要点

本实例的关键点是 while()循环语句的灵活运用。根据提交的参数值，执行 while()循环语句，在循环体中执行删除操作，完成数据的循环删除操作。

while()循环语句，其作用是反复地执行某一项操作，是循环控制语句中最简单的一个，




也是最常用的一个。while()循环语句对表达式的值进行判断，当表达式为非0值时，执行while()语句中的内嵌语句；当表达式的值为0值时，则不执行while()语句中的内嵌语句。该语句的特点是：先判断表达式，后执行语句。while()循环控制语句的操作流程如图3.4所示。

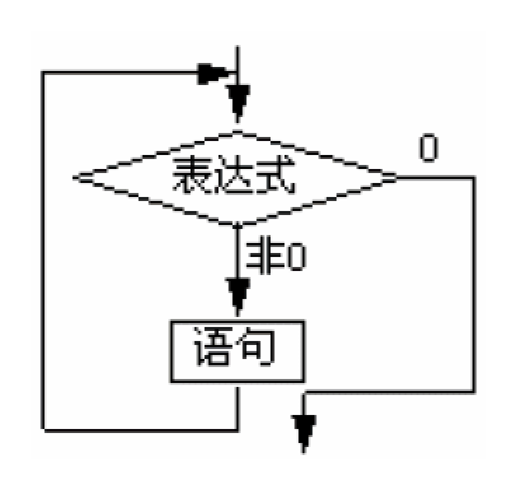


图3.4　while()循环控制语句的操作流程

其语法如下：

```
while (expr){                                   /*
        statement;                   先判断条件，当条件满足时执行语句块否则
                                              不向下执行
}                                               */
```

只要while表达式expr的值为true，就重复执行嵌套中的statement语句，如果while表达式的值一开始就是false，则循环语句一次也不执行。

实例037　do…while语句循环读取数据库中数据

（实例位置：配套资源\SL\03\037　视频位置：配套资源\SP\03\037）

实例说明

本实例通过do…while循环语句循环读取数据库中的数据，并输出员工详细信息，运行结果如图3.5所示。

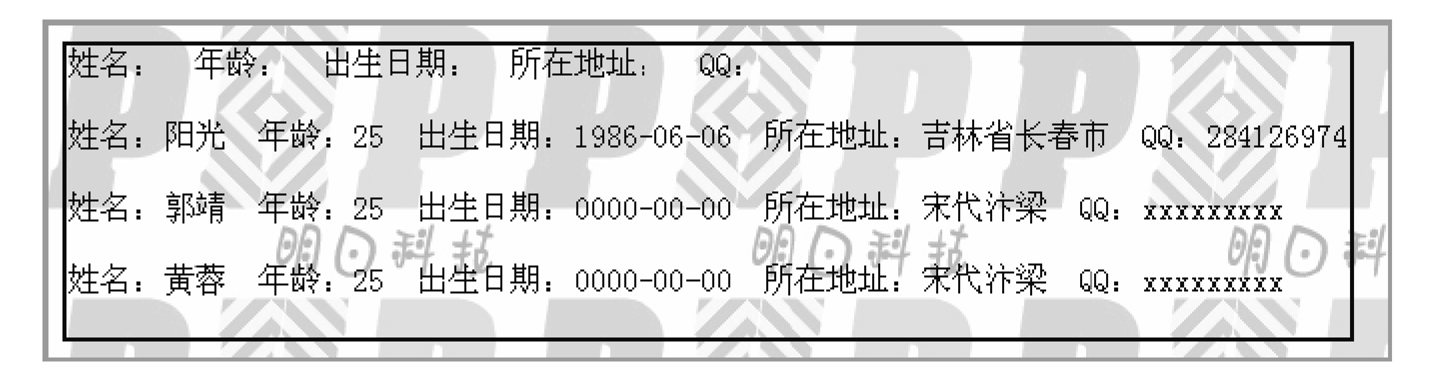


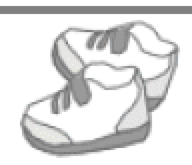
图3.5　员工详细信息浏览

实现过程

具体步骤如下：

（1）创建PHP脚本文件。首先，利用mysql_connect函数连接MySQL数据库，利用mysql_select_db函数选择数据库，并通过mysql_query函数定义页面的编码格式。然后，定义SQL查询语句并返回结果集。最后，通过do…while语句循环输出数据库中的数据。其代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111")          //连接数据库服务器
  mysql_select_db("db_database03",$conn);                  //连接数据库
  mysql_query("SET NAMES GBK");                            //设置编码格式
  $rs = mysql_query("select * from tb_while");             //执行查询
  do{
```





```
            echo "姓名：".iconv('gbk','utf-8',$rst[1])."  年龄：".iconv('gbk','utf-8',$rst[2])."  出生日期： ".iconv('gbk','utf-8',$rst[3])."  所在地址： ".iconv('gbk','utf-8',$rst[4])."  QQ： ".iconv('gbk','utf-8',$rst[5])."<br><br>";
        }while($rst = mysql_fetch_array($rs));
    ?>
```

（2）将该文件存储于\MR\03\037\文件夹下，并命名为 index.php。运行结果如图 3.5 所示。

脚下留神：

在使用 do…while 语句之前，要事先考虑程序是否有必要在判断条件之前运行一次。如果没必要，就尽量不要使用；否则，可能会出现意外的输出结果。

技术要点

while 语句还有另外一种表示形式——do…while。do…while 循环语句和 while 循环语句非常类似，只是 do…while 循环语句是在循环底部检测循环表达式，而不是在循环的顶部进行检测。do…while 循环语句的语法格式如下：

```
do{
        statment
}while(expr);
```

该语句的操作流程是：先执行一次指定的循环体语句，然后判断表达式的值。当表达式的值为非 0 时，返回重新执行循环体语句。如此反复，直到表达式的值等于 0 为止，此时循环结束。其特点是先执行循环体，然后判断循环条件是否成立。do…while 循环语句的操作流程如图 3.6 所示。

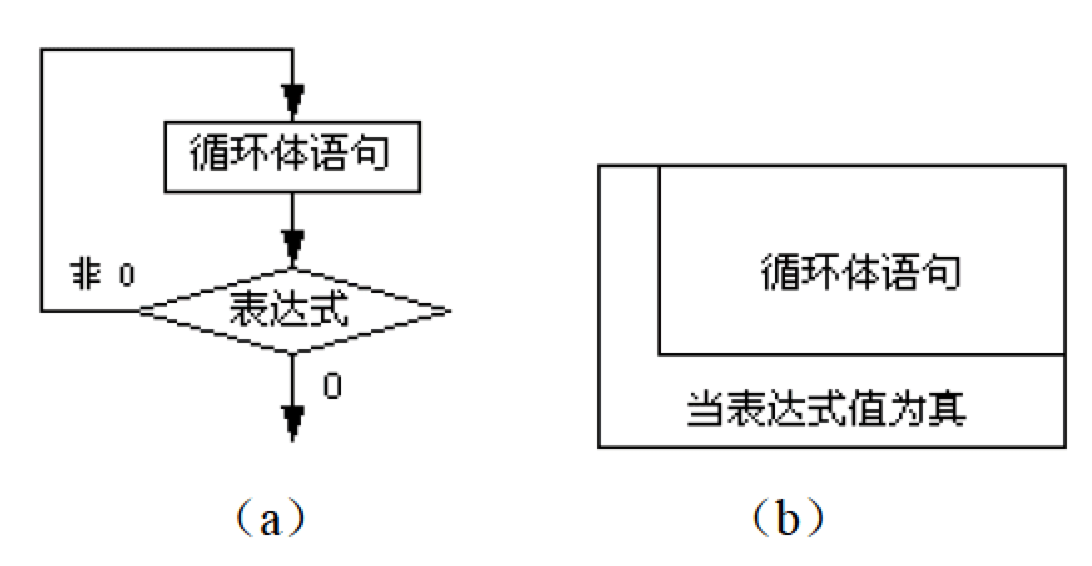


图 3.6 do…while 循环语句的操作流程

实例 038 生成随机验证码

（实例位置：配套资源\SL\03\038 视频位置：配套资源\SP\03\038）

实例说明

在用户登录页面或留言发表页面经常会看到验证码的身影，通过这项技术可以很大程度地提高网站的安全性。运行本实例，可以发现在如图 3.7 所示的登录页面中随机产生了 4 位数字（6191）。每次刷新页面，这 4 位数字都在发生改变。用户登录时要求输入这 4 位





Note

数字，这样可以防止用户通过恶意程序来试探登录密码的值从而非法登录网站。



图 3.7　生成随机验证码

实现过程

具体步骤如下：

（1）创建用户登录页面，添加用户登录表单。

（2）创建 JavaScript 脚本，定义 chkuserinput()方法对表单中提交的数据进行验证。

```
<script language="javascript">
  function chkuserinput(form){
        if(form.username.value==""){
              alert("请输入用户名!");
              form.username.select();
              return(false);
        }
        if(form.userpwd.value=="") {
              alert("请输入用户密码!");
              form.userpwd.select();
              return(false);
        }
        if(form.yz.value==""){
              alert("请输入验证码!");
              form.yz.select();
              return(false);
        }
        return(true);
  }
</script>
```

（3）编写 PHP 脚本，通过 mt_rand()函数生成 4 位随机验证码，并输出 4 位随机验证码对应的图片。运行结果如图 3.7 所示。

技术要点

应用 PHP 的随机函数 mt_rand()生成随机验证码，并根据随机验证码的值读取对应的数字图片生成验证码图像。

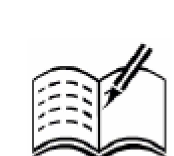
mt_rand()函数用于产生 min 和 max 之间的随机数。其语法如下：

```
int mt_rand([int min],[int max])
```



参数 min 和 max 指定随机数的取值范围。

在读取生成的随机验证码时，应用到 for 语句和 substr()函数对验证码的值进行读取，并根据读取的值定义输出的数字图像。其关键代码如下：



```
<?php
        $num = intval ( mt_rand ( 1000, 9999 ) );                    //生成随机验证码
        for($i = 0; $i < 4; $i ++) {                                 //循环读取验证码
            echo "<img src=images/code/" . substr ( strval ( $num ), $i, 1 ) . ".gif>";//输出数字图像
    }
?>
```

实例 039　图形计数器

（实例位置：配套资源\SL\03\039　视频位置：配套资源\SP\03\039）

实例说明

网站计数器的形式有很多，图形计数器是比较常用的一种。PHP 制作图形计数器的方法主要有两种：一种可以利用 GD2 函数直接绘制图形，另一种可以事先用图形绘制工具将图形绘制出来，然后用 PHP 代码进行调用。本实例将采用第二种方式来实现统计网页的访问量，运行结果如图 3.8 所示。

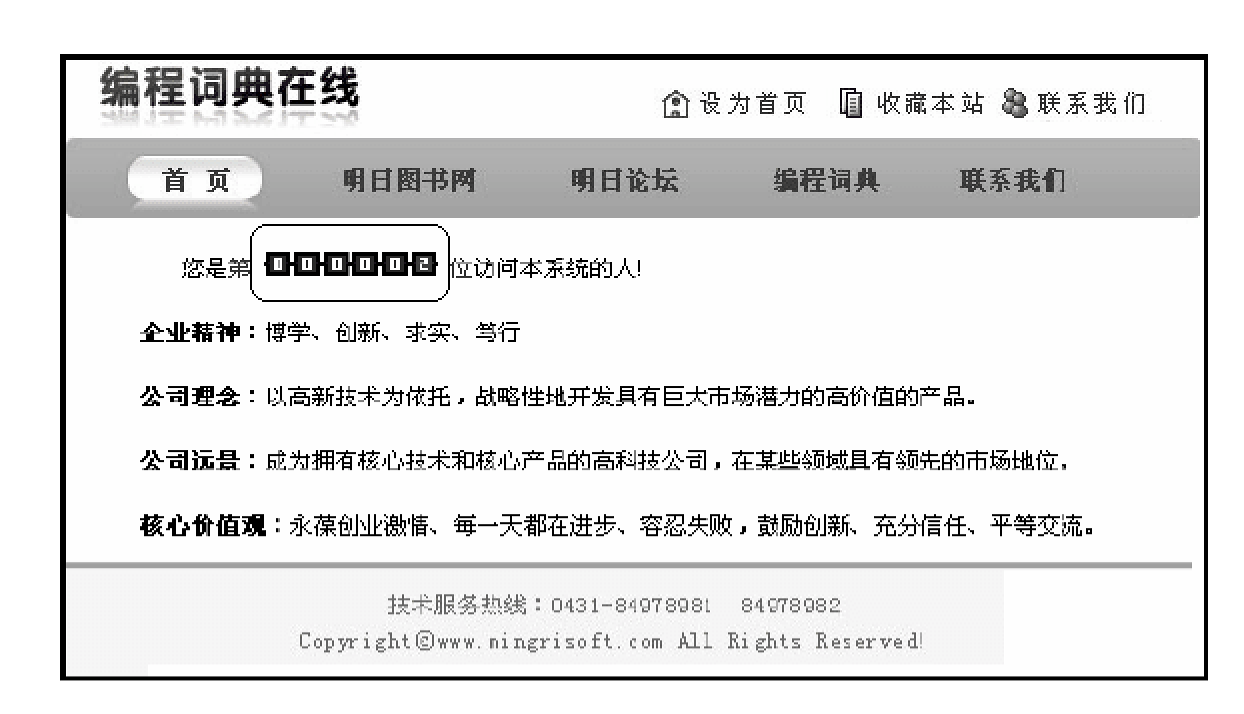


图 3.8　图形计数器

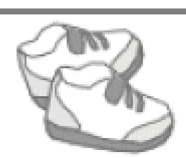
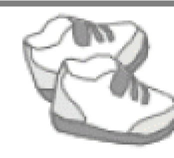
实现过程

具体步骤如下：

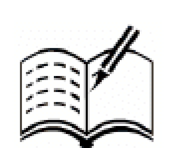
（1）创建 index.php 文件，设计网站的页面，输出公司简介的内容。

（2）在 index.php 文件中，通过文件系统函数编写一个文本计数器，将数据存储于文本文件 counter.txt 中，并通过 SESSION 变量屏蔽页面的刷新。其关键代码如下：

```
<?php
session_start ();
if ($_SESSION [temp] == "") { //判断$_SESSION[temp]==""的值是否为空,其中的 temp 为自定义
的变量
    if (($fp = fopen ( "counter.txt", "r" )) == false) {
        echo "打开文件失败!";
    } else {
        $counter = fgets ( $fp, 1024 );              //读取文件中数据
        fclose ( $fp );                              //关闭文本文件
        $counter ++;                                 //计数器增加 1
        $fp = fopen ( "counter.txt", "w" );          //以写的方式打开文本文件
        fputs ( $fp, $counter );                     //将新的统计数据增加 1
```

```
            fclose ( $fp );                              //关闭文件
        }
        $_SESSION [temp] = 1;                            //为$_SESSION[temp]赋一个值
    }
    ?>
```



Note

（3）在 index.php 文件中，通过文件系统函数读取文本文件中存储的网站访问量的数据，并将数据以图形的形式输出。相关代码请参考技术要点中的内容。

多学两招：

第二种将数字转换成图形的方法如下：

首先，应用函数 intval()将计数器的值转变成字符串。然后设置计数器最多显示 8 位数字（当然位数可以根据实际情况设定），并根据 strlen()函数确定当前计数值的位数，再用 8 减去该位数值，得出剩余的位数（为了使页面美观，剩余的位数值用 0 填充）。最后输出文本文件中统计的数据。其关键代码如下：

```
$imagenum=intval($num);                                  //类型转换
for($i=0;$i<8-strlen($imagenum);$i++){
    echo "<img src=images/count/0.gif>";                 //用 0 填充剩余的位数
}
for($i=0;$i<strlen($imagenum);$i++){                     //通过循环调用图片显示结果
    echo "<img src=images/count/".substr($imagenum,$i,1).".gif>";
}
```

技术要点

图形计数器设计原理如下：

（1）为了避免用户通过刷新页面来提高网站的访问量，可以通过 SESSION 变量进行屏蔽。首先，通过判断$_SESSION [temp]的值是否为空决定计数器的值是否加 1，如果该值不为空，说明用户正在访问该网站，这时计数器的值不加 1，反之加 1。

（2）将文本文件中存储的访问量数据以数字图片的形式输出。首先，应用 strlen()函数获取文本文件中数据的长度。然后，定义网站访问量的最大值为 6 位数字（当然位数可以根据实际情况设定），用 0 填充剩余位数。最后，通过 for 循环和 switch 语句，将从文本文件中读取的数据以图形的形式输出，其关键代码如下：

```
        $len = strlen ( $counter );                          //获取字符串的长度
        $str = str_repeat ( "0", 6 - $len );                 //获取 6-$len 个数字 0
        for($i = 0; $i < strlen ( $str ); $i ++) {           //获取变量$str 的字符串长度
            $result = $str [$i];
            $result = '<img src=images/0.gif>';
            echo $result;                                    //循环输出$result 的结果
        }
    for($i = 0; $i < strlen ( $counter ); $i ++) {           //获取字符串的长度
            $result = $counter [$i];
            switch ($result) {
                //如果值为"0",则输出 0.gif 图片
                case "0" :
```






```
                    $ret [$i] = "0.gif";
                    break;
               case "1" :
                    $ret [$i] = "1.gif";
                    break;
               case "2" :
                    $ret [$i] = "2.gif";
                    break;
          …//省略部分代码
          }
          echo "<img src=images/" . $ret [$i] . ".>";              //输出文本文件中存储的数据
     }
```

实例 040　包含数据库连接文件

（实例位置：配套资源\SL\03\040　视频位置：配套资源\SP\03\040）

实例说明

一个程序可能要与数据库多次交互，所以数据库连接信息要单独保存在一个文件中，这也是代码重用的一种体现。本实例通过 include()语句包含数据库连接文件，运行结果如图 3.9 所示。

编号：1　姓名：杨明
编号：2　姓名：潘凯华
编号：3　姓名：杨明
编号：4　姓名：潘凯华
编号：5　姓名：刘中华
编号：6　姓名：刘中华

图 3.9　包含数据库连接文件

实现过程

具体步骤如下：

（1）创建 index.php 文件，通过 include()语句包含数据库文件，并执行查询操作，输出结果。其代码如下：

```
<?php
include ("inc.php");                                   //包含数据库连接文件
$sql = "select * from tb_include";                     //sql 语句
$rs = mysql_query($sql);                               //执行查询操作
while($rst = mysql_fetch_array($rs)){                  //将查询结果循环输出
          echo '编号：'.$rst[0].'  姓名：'.$rst[1]."<br>";
}
?>
```

（2）创建数据库连接文件 inc.php。其代码如下：

```
<?php
$conn = mysql_connect("localhost","root","111");       //连接 MySQL
mysql_select_db("db_database03",$conn);                //连接数据库
mysql_query("SET NAMES GBK");                          //设置编码格式
?>
```

技术要点

include()语句与 require()语句在作用上是完全相同的，但在使用 include 语句引用外部



文件时，只有代码执行到 include 语句，才将外部文件引用进来并读取文件的内容，当所引用的外部文件发生错误时，系统只给出一个警告，整个 PHP 文件则继续向下执行。include 语句的语法如下：



Note

```
void include(string filename);
```

参数说明：

filename：指定的完整路径文件名。

实例 041 健康生活提醒

（实例位置：配套资源\SL\03\041 视频位置：配套资源\SP\03\041）

实例说明

一些手机在开机时会出现开机问候语或者提示今日日程。本实例就是类似这样的程序，通过 switch()语句根据当前日期给出健康生活提示信息，运行结果如图 3.10 所示。

今天是黑色星期五。。。。。。

图 3.10 健康生活提醒

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，通过 date 函数取得当前是星期几。然后，利用 switch 语句进行操作，并通过 echo 语句输出信息提示。其代码如下：

```
<?php
    $a = date("l");                                      //取得当前是星期几
    switch($a){
        case "Monday";                                   //当前如果是星期一
        echo "今天是星期一，一周忙碌生活开始了";         //输出
        break;
        case "Tuesday";                                  //当前如果是星期二
        echo "今天是星期二，电视台下午两点以后部分休息";//输出
        break;
        case "Wednesday";                                //当前如果是星期三
        echo "今天是星期三，下午有乒乓球比赛";           //输出
        break;
        case "Thursday";                                 //当前如果是星期四
        echo "今天是星期四，晚上有 NBA 的重播";          //输出
        break;
        case "Friday";                                   //当前如果是星期五
        echo "今天是黑色星期五……";                     //输出
        break;
        case "Saturday";                                 //当前如果是星期六
        echo "今天是星期六，明天就放假了哈哈哈";         //输出
        break;
        case "Sunday";                                   //当前如果是星期日
```



```
                echo "今天是星期天，可以玩上一整天";                //输出
            }
    ?>
```

（2）将该文件存储于\MR\03\041\文件夹下，并命名为 index.php。运行结果如图 3.10 所示。



指点迷津：

switch()语句与 if…else…语句在功能上基本相同，只是 switch()语句更加灵活，格式更加鲜明，结构更加清晰。

技术要点

本实例的关键点是利用 switch 语句，将地址栏传递的数据与 switch 中 case 值进行比较，符合条件的利用 echo 语句输出对应数据。

虽然 elseif 语句可以进行多重选择，但使用上十分烦琐。为了避免 if 语句的冗长，提高程序的可读性，可以使用 switch 分支控制语句。switch 语句的语法格式如下：

```
switch(variable){
  case value1:
        statement1;
        break;
  case value2:
  …
  default:
        defulat statement;
}
```

switch 语句根据 variable 的值，依次与 case 中的 value 值相比较。如果不相等，则继续查找下一个 case；如果相等，就执行对应的语句，直到 switch 语句结束或遇到 break 为止。一般 switch 语句最终都有一个默认值 default，如果在前面的 case 中没有找到相符的条件，则输出默认语句，这与 else 语句类似。

switch 语句的流程控制如图 3.11 所示。

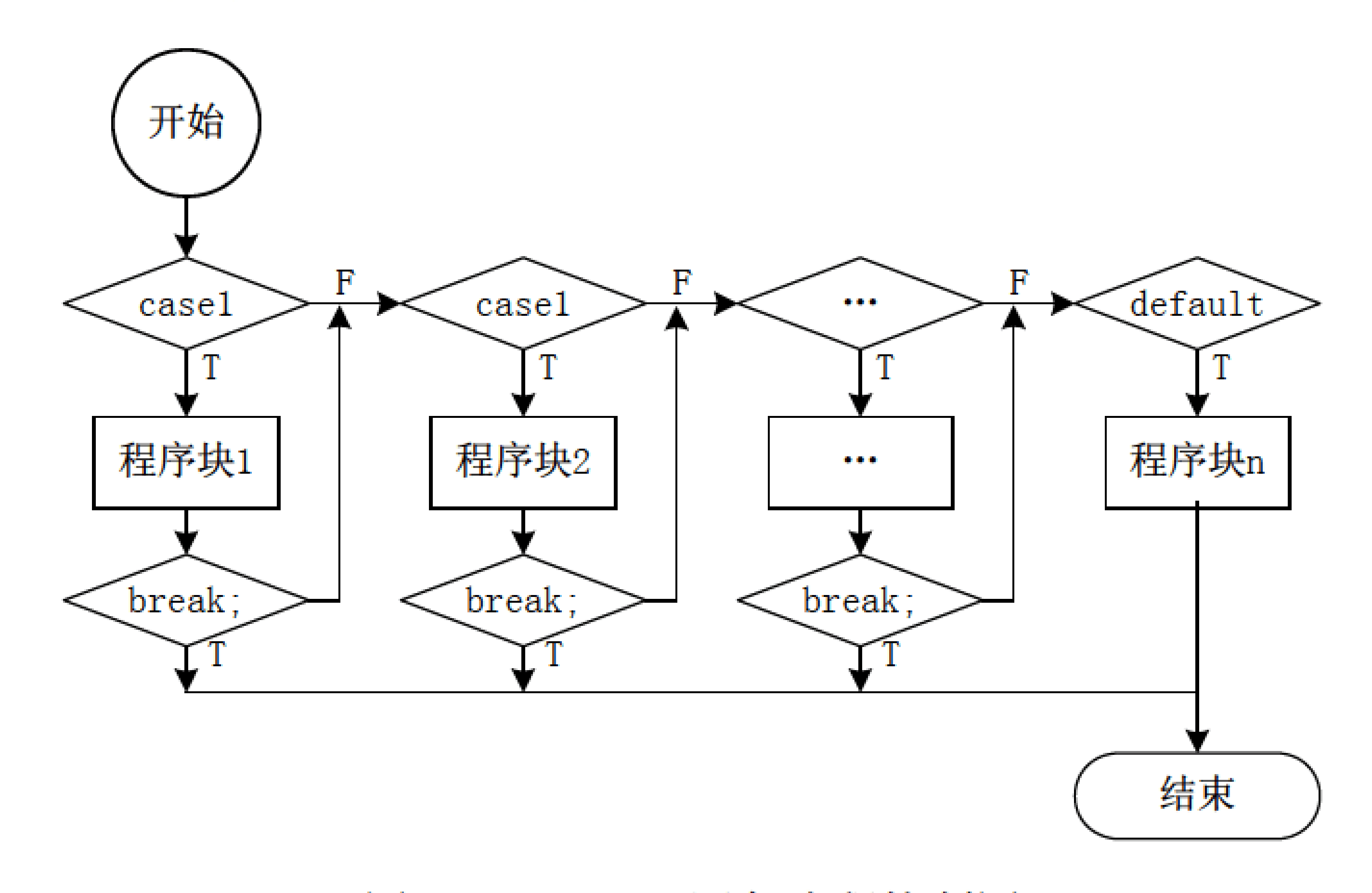


图 3.11　switch 语句流程控制图



实例 042　if 语句判断美女征婚条件

（实例位置：配套资源\SL\03\042　视频位置：配套资源\SP\03\042）



实例说明

条件语句的应用，小到一个简单的判断登录用户权限的模块，大到高负载、高访问量的大型网站的整个程序中都会有它的身影。如何熟练地掌握 if 条件语句的应用呢？其实很简单，每个应用语句都有其特点和使用原则，特别是在多层嵌套的 if 语句中，它的应用原则是：无论多少层语句的嵌套，只要找到 if 语句开始处和闭合处，一般就不会产生逻辑错误。

本实例来源于网络上的一个笑话，其主要内容为：一女子让计算机为其征婚，开出征婚条件有两点：① 要帅；② 要有车。计算机为她搜寻到了结果并回答她说：女士您搜寻的是象棋吗？这位女子不愿接受此次搜寻的结果又重新输入条件：① 要有很多钱；② 要有漂亮的房子。计算机为她再次搜寻了结果：女士，您搜寻的是银行吗？此女子仍然不失望，继续输入条件：① 要有安全感；② 要长得酷。结果计算机搜出的是：女士，您搜寻的是奥特曼吗？此女子想尝试最后一次，于是输入条件：① 要有很多钱、要长得酷、要有安全感；② 有车、要帅、要有漂亮的房子。计算机搜寻了很久才回答道：哦，女士，我知道了，您搜寻的是奥特曼在银行里下象棋。程序的运行结果如图 3.12 所示。

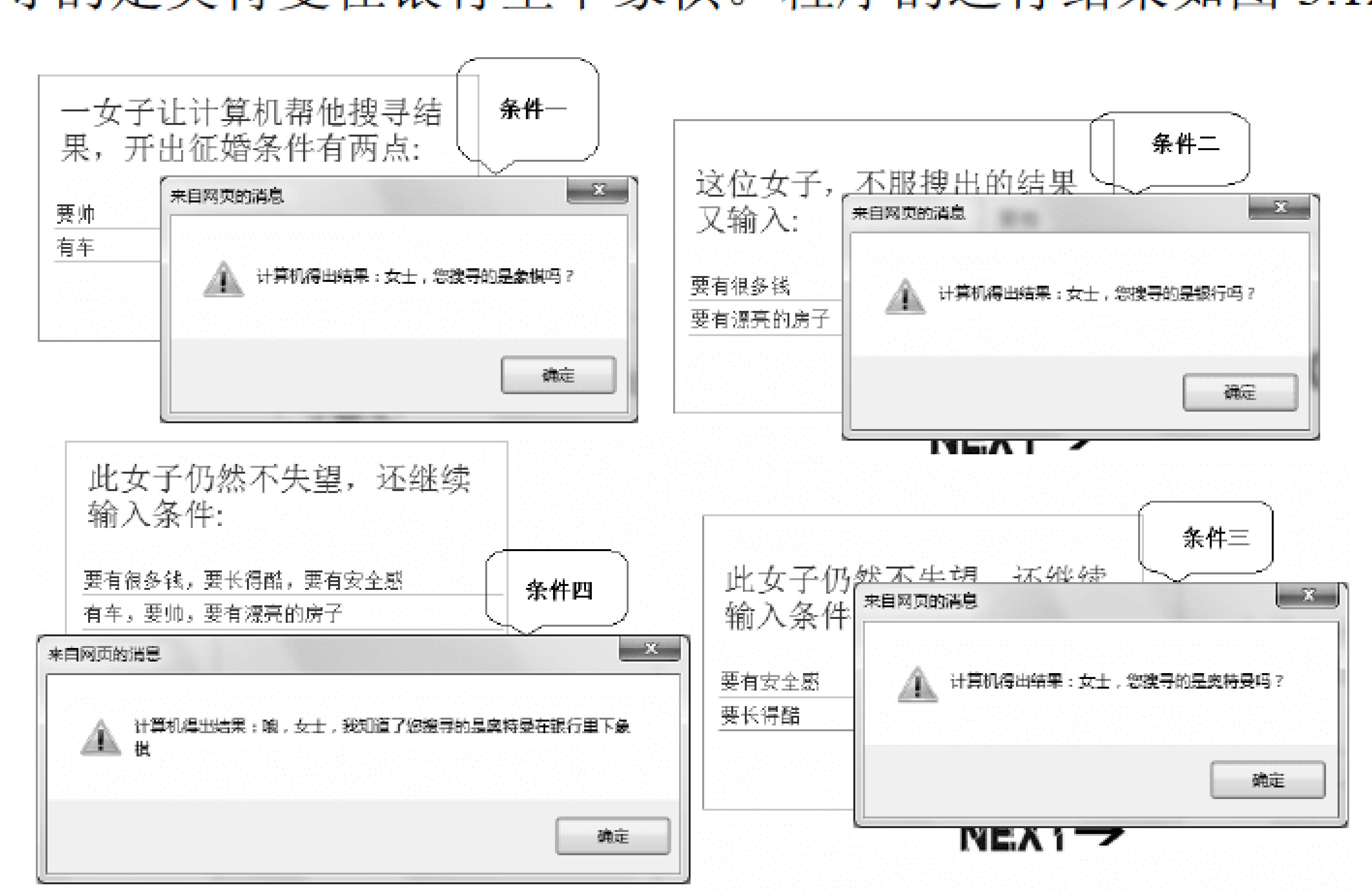


图 3.12　if 语句判断美女征婚条件

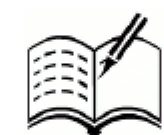
实现过程

具体步骤如下：

（1）自定义函数，通过定义页面的布局，动态地向页面中插入文本和提交按钮的 value 属性值。其具体方法是：首先定义两个<div>标签，在内层的<div>标签中定义<p>标签，利用插入 PHP 代码的形式动态接收自定义函数首部$str 参数。提交按钮<input>标签中，在 value 属性值的位置也使用同样的方法接收参数。其具体代码如下：

```
<?php
function if_isset($str,$str1){
```





```
?>
<div class="one">
        <div id="one">
            <p><?php echo $str;?></p>
            <form action=""method="post">
                <input type="text" name="text1"size="40" value="要帅" >
                <input type="text" name="text" size="40" value="有车" >
                <input class="two" type="submit" name="<?php echo $str1?>" value="">
            </form>
        </div>
</div>
<?php
 }
 if(!isset($_POST['sub'])&&!isset($_POST['sub1'])&&!isset($_POST['sub2'])){
     if_isset("一女子让计算机帮她搜寻结果，开出征婚条件有两点:","sub");
 }
 if(isset($_POST['sub'])){
     if_isset("这位女子，不服搜出的结果又输入:","sub1");
 }
 if(isset($_POST['sub1'])){
     if_isset("此女子仍然不失望，继续输入条件:","sub2");
 }
 if(isset($_POST['sub2'])){
     if_isset("此女子仍然不失望，还继续输入条件:","sub3");
 }
```

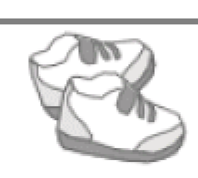
（2）利用 if...else 语句和 if...elseif 语句嵌套的形式，根据接收文本框参数的不同，输出对应的页面信息。其代码如下：

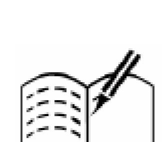
```
if(isset($_POST['sub'])||isset($_POST['sub1'])||isset($_POST['sub2'])||isset($_POST['sub3'])){
    if($_POST['text']=="有车"&&$_POST['text1']=="要帅"){
        echo "<script>alert('计算机得出结果：女士，您搜寻的是象棋吗？')</script>";
    }elseif($_POST['text']=="要有漂亮的房子"&&$_POST['text1']=="要有很多钱"){
        echo "<script>alert('计算机得出结果：女士，您搜寻的是银行吗？')</script>";
    }elseif($_POST['text']=="要长得酷"&&$_POST['text1']=="要有安全感"){
        echo "<script>alert('计算机得出结果：女士，您搜寻的是奥特曼吗？')</script>";
    }elseif($_POST['text']=="有车，要帅，要有漂亮的房子"&&$_POST['text1']=="要有很多钱，
要长得酷，要有安全感"){
        echo "<script>alert('计算机得出结果：哦，女士，我知道了，您搜寻的是奥特曼在银
行里下象棋');location.href='index.php'</script>";
    }else{
        echo "<script>alert('输入信息错误');</script>";
    }
}
?>
```

技术要点

本实例主要应用条件判断语句 if...else 语句和 if...elseif 语句。

大多数时候，总是需要在满足某个条件时执行一条语句，而在不满足该条件时执行其





Note

他语句。这时，可以使用 else 语句，其语法格式为：

```
if(expr){
  statement1;
}else{
  statement2;
}
```

该语句的意思为：当表达式 expr 为真时，执行 statement1；如果表达式 expr 为假，则执行 statement2。if 语句的流程图如图 3.13 所示。

if…else 语句只能选择两种结果：要么执行真，要么执行假。但现在有两种以上的选择该怎么办呢？例如，一个班的考试成绩，如果在 90 分以上，则为“优秀”；如果在 60～90 分之间，则为“良好”；如果低于 60 分，则为“不及格”。这时，可以使用 elseif（也可以写做 else if）语句来执行。该语法格式为：

```
if(expr1){
  statement1;
}else if(expr2){
  statment2;
}…
else{
  statementn;
}
```

elseif 语句的流程图如图 3.14 所示。

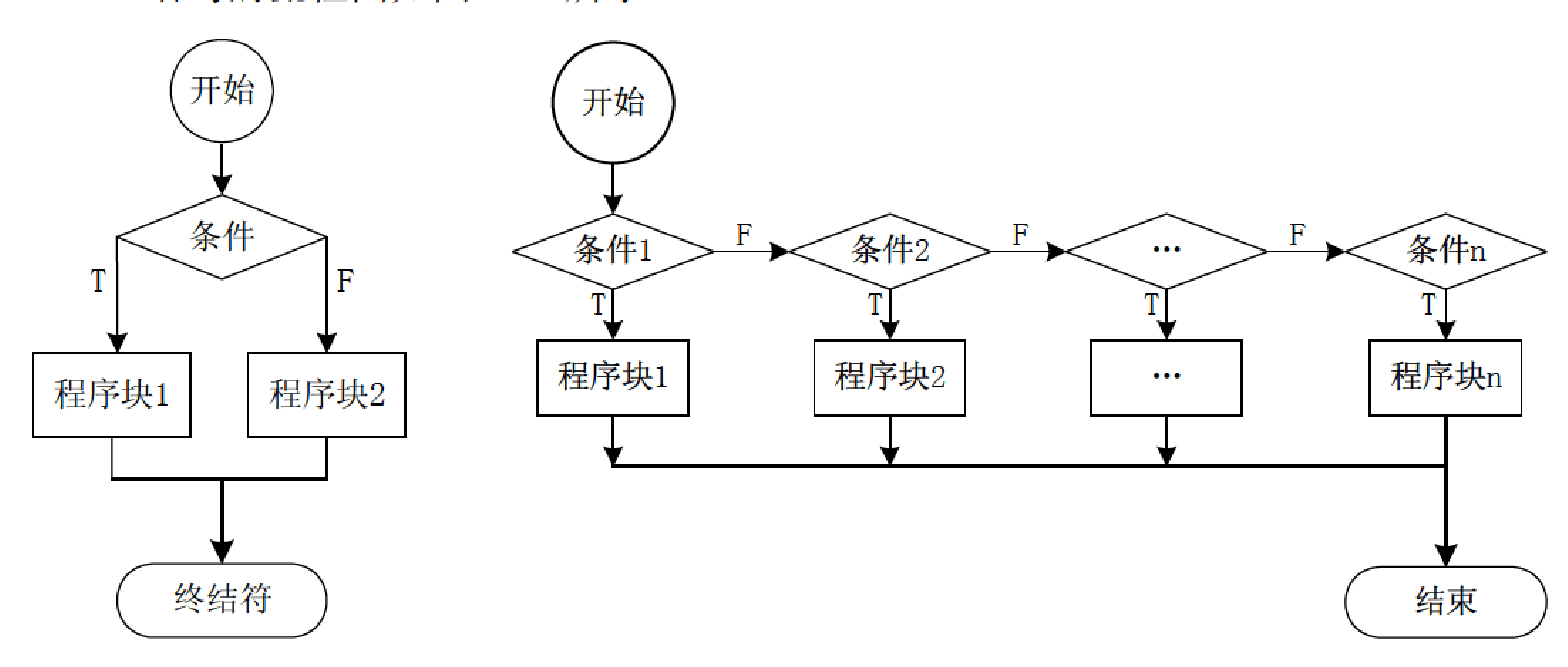


图 3.13　if…else 语句流程控制图　　　　图 3.14　elseif 语句的流程控制图

实例 043　网页版九九乘法表

（实例位置：配套资源\SL\03\043　视频位置：配套资源\SP\03\043）

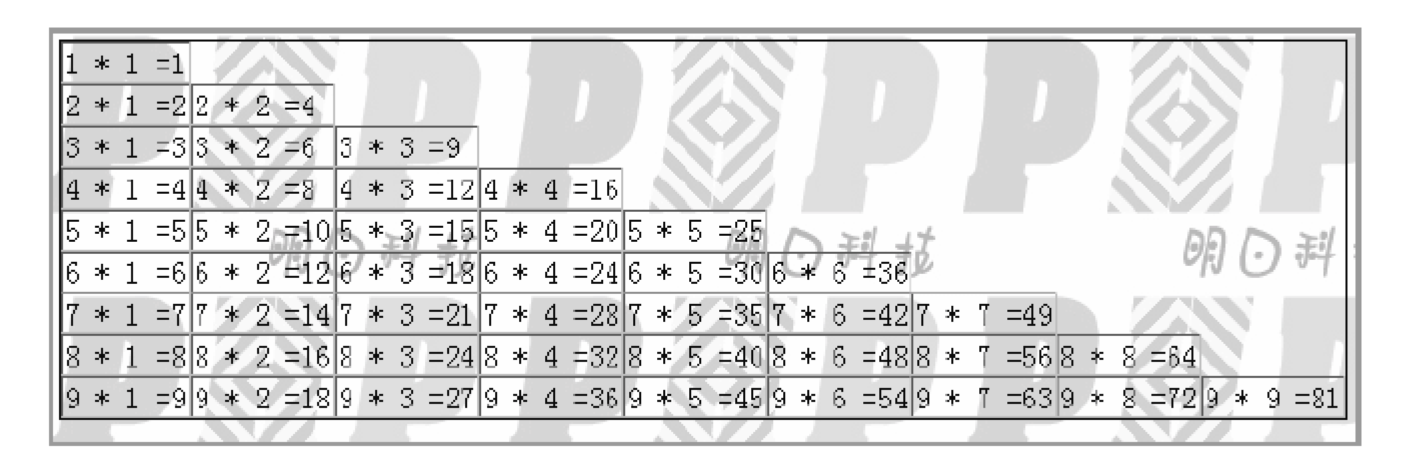
实例说明

在编程世界里，算法是一门独立于语法、函数之外的知识。它着重考验用户的逻辑思





维能力并且与数学知识息息相关。本实例通过 for()循环语句实现网页版九九乘法表，运行结果如图 3.15 所示。



1 * 1 =1								
2 * 1 =2	2 * 2 =4							
3 * 1 =3	3 * 2 =6	3 * 3 =9						
4 * 1 =4	4 * 2 =8	4 * 3 =12	4 * 4 =16					
5 * 1 =5	5 * 2 =10	5 * 3 =15	5 * 4 =20	5 * 5 =25				
6 * 1 =6	6 * 2 =12	6 * 3 =18	6 * 4 =24	6 * 5 =30	6 * 6 =36			
7 * 1 =7	7 * 2 =14	7 * 3 =21	7 * 4 =28	7 * 5 =35	7 * 6 =42	7 * 7 =49		
8 * 1 =8	8 * 2 =16	8 * 3 =24	8 * 4 =32	8 * 5 =40	8 * 6 =48	8 * 7 =56	8 * 8 =64	
9 * 1 =9	9 * 2 =18	9 * 3 =27	9 * 4 =36	9 * 5 =45	9 * 6 =54	9 * 7 =63	9 * 8 =72	9 * 9 =81

图 3.15　网页版九九乘法表

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，定义一层循环中的循环变量为$a，设置$a 的初始值为 1，最大值为 9，并且每执行一次循环就做一次自增运算。然后，定义内层循环中的循环变量$b，设置$b 的初始值为 1，最大值小于$a，并且做自增运算，最后，将 2 层循环控制的数据输出到页面。其代码如下：

```
<?php
        for($a = 1;$a < 10;$a++){                          //1 层循环
            echo "<tr>";
            for($b = 1;$b <= $a;$b++){                     //2 层循环控制 1 层循环输出
                echo "<td>";
                echo "$a * $b =".$a*$b;                    //输出结果
                echo"</td>";
            }
            echo "</tr>";
        }
?>
```

（2）将该文件存储于\MR\03\043 文件夹下，并命名为 index.php。运行结果如图 3.15 所示。

指点迷津：

算法不是一朝一夕就可以掌握的，它是一个慢慢了解并形成思想的过程。

技术要点

本实例的关键点是利用 for 循环中，第二层循环控制第一层循环实现指定数据的输出。for 循环是 PHP 中最复杂的循环结构，其语法格式为：

```
for (expr1; expr2; expr3){
statement;
}
```

其中：expr1 在第一次循环时无条件取一次值。expr2 在每次循环开始前求值。如果值为真，则执行 statement；否则，跳出循环，继续往下执行。expr3 在每次循环后被执行。for 循环语句的流程控制图如图 3.16 所示。





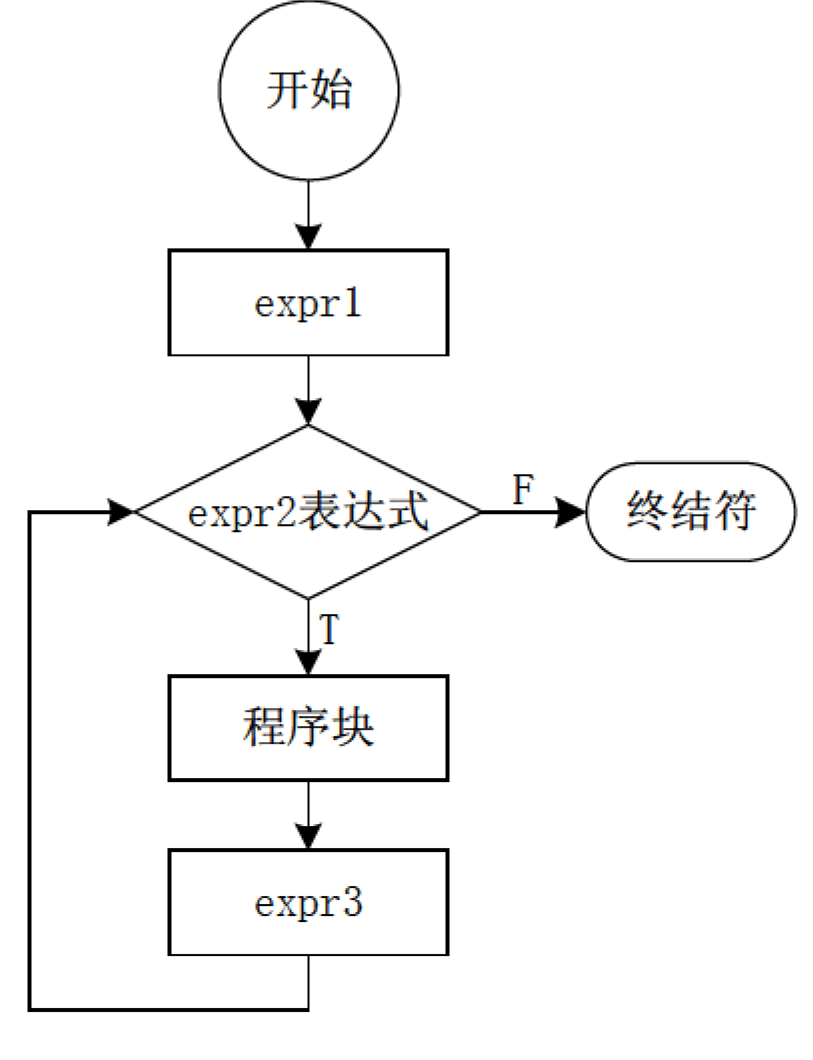


图 3.16　for 循环语句流程控制图

实例 044　读取购物车中的数据

（实例位置：配套资源\SL\03\044　视频位置：配套资源\SP\03\044）

实例说明

用户在进行数据库开发中，小型的数据没有必要与数据库进行交互，可以直接将数据存入数组，这样不仅可以节约开发时间，还可以节省服务器资源。本实例通过对数组函数的相关操作，实现读取数组购物车中的数据，运行结果如图 3.17 所示。

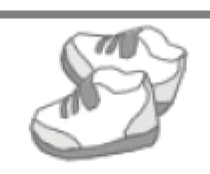


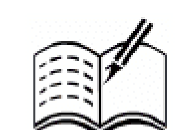
图 3.17　读取购物车中的数据

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，定义字符串变量，并且在变量中插入“#”作为字符串分割点。然后，利用 GET 方式接收地址栏传递的参数。当$_GET[push]的值等于 1 时，输出 JavaScript 成功提示。当$_GET[pop]等于 1 时，利用字符串拆分函数将字符串变量以#分割，并分别保存到数组之中。最后，利用 foreach 语句将字符串数组中的 value 值输出到





页面中。其代码如下：

```
<?php
        $str = "宝贝：液晶电脑显示器#一口价：900 元#运费：30 元（自理）";        //定义字符串
            if($_GET[push] == 1){                                          //通过 get 方式取得参数
                echo "<script>alert('成功放入购物车');</script>";              //提示消息
            }
            if($_GET[pop] == 1){                                           //通过 get 方式取得参数
                $strexplode = explode("#",$str);                            //分割字符串
                foreach($strexplode as $key => $value){                    //遍历数组
?>
                <tr><td><?php echo $value;?></td></tr>                     //输出结果
<?php
                }
            }
?>
```

（2）将该文件存储于\MR\03\044 文件夹下，并命名为 index.php。运行结果如图 3.17 所示。

技术要点

本实例的关键点是运用 foreach 语句遍历数组，取得 key 值和对应的 value 值。foearch 循环在 PHP 4.0 中，其擅长处理数组，是遍历数组的一种简单方法。在 PHP 5.0 中，新增加了对对象的支持。该语句的语法格式如下：

```
foreach (array_expression as $value)
statement
```

或

```
foreach (array_expression as $key => $value)
    statement
```

forearch 语句将遍历数组 array_expression，每次循环时，将当前数组中的值赋给$value（或是$key 和$value）。同时，数组指针向后移动，直到遍历结束。当使用 foreach 语句时，数组指针将自动被重置，所以不需要手动设置指针位置。

实例 045　多图片上传

（实例位置：配套资源\SL\03\045　视频位置：配套资源\SP\03\045）

实例说明

上传图片是很多娱乐网站所必须包含的模块，其优点是可良好地与用户互动，由用户选择喜欢的图片或者文件上传。大量的用户为网站传递数以千计的文件和图片，这对于一个企业网站来说，是一个相当可观的收益。简单的单文件上传实现起来相当简单，这里笔者使用 for 循环语句，为图片上传增加了一些扩展——多文件上传。本实例应用 for 循环控制语句实现将多图片上传到服务器指定文件夹下，一次可以上传 4 张以内的图片，运行结果如图 3.18 所示。







图 3.18　多图片上传

实现过程

具体步骤如下：

（1）创建 form 表单，添加 4 个文件域和“确定上传”按钮，将数据通过 POST 方法提交到指定的位置。

（2）通过$_POST[]全局数组获取表单中提交的数据，应用 for 循环获取表单中提交的数据，并通过$_FILES 全局数组和 move_upload_file()函数完成文件的上传操作。其关键代码如下：

```
<?php
 if(isset($_POST[sub])){                              //如果页面中存在变量$_POST[sub]程序向下执行
     for($a=1;$a<4;$a++){                                       //for 循环语句
         $name=$_FILES['file_'.$a];                             //将图片信息保存在变量中
         $rand=rand(1,100000);                                  //获取 1～100000 的随机数
         $name_type=substr($name['name'],-4,4);//截取图片名称的后四位数据并保存在变量中
         $time=microtime();                                     //获取时间戳微秒数
         $path='upfiles/'.(($rand+$time).$name_type); //上传路径
         $dir='upfiles/';                                       //文件夹名称
         $move = move_uploaded_file($name['tmp_name'],$path);//将图片移动到指定文件夹下
         if($move==true){
             echo "成功上传文件".($rand+$time).$name_type."<br>";
         }
     }
 }
 ?>
```

技术要点

开发本实例，需要对文件系统的知识有初步的了解，其中应用到$_FILE 数组和 move_upload_file()函数。move_upload_file()函数的语法格式如下：

```
bool move_uploaded_file(string filename, string destination)
```

move_uploaded_file()函数的参数说明如表 3.1 所示。

表 3.1　move_uploaded_file()函数参数的说明

参　　数	说　　明
filename	必选参数。指定要上传的文件地址
destination	必选参数。上传到服务器后的存储目录及名称





脚下留神：

应用 POST 方法上传文件时，应当在上传表单的<form>标记中添加以下内容：enctype="multipart/form-data"。

实例 046　控制页面中表情图的输出

（实例位置：配套资源\SL\03\046　视频位置：配套资源\SP\03\046）

实例说明

break 语句的含义是结束当前 for、foreach、while、do…while 或者 switch 结构的执行。本实例通过 break 语句实现控制页面中表情图的输出，运行结果如图 3.19 所示。

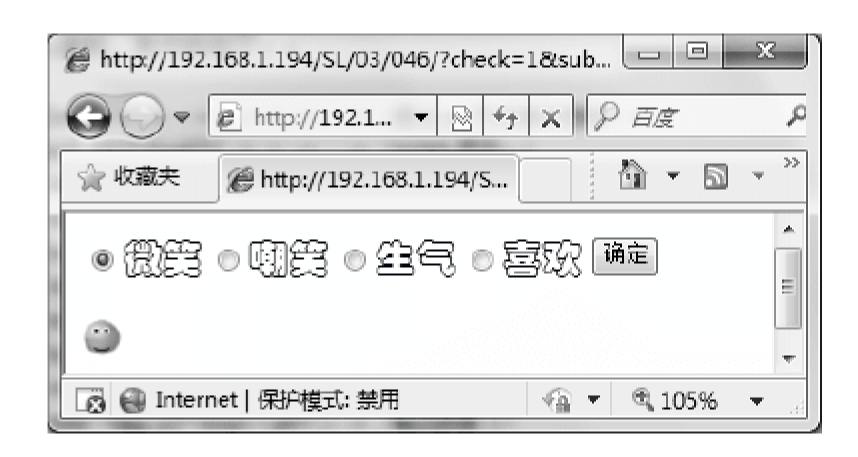


图 3.19　控制页面中表情图的输出

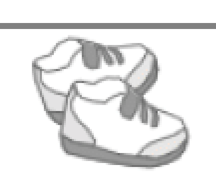
实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。当单击“确定”按钮时，利用 switch 语句选择输出的图片，并通过 break 语句跳出当前 switch 语句。其代码如下：

```
<?php
	if($_GET[sub]){                                   //通过 get 方式接收参数
		switch($_GET['check']){                       //switch 语句
			case "1";                                 //如果值为 1
			echo "<img src='image/1.gif>";            //输出图像 1
			break;                                    //跳出语句
			case "2";                                 //如果值为 2
			echo "<img src='image/2.gif>";            //输出图像 2
			break;                                    //跳出语句
			case "3";                                 //如果值为 3
			echo "<img src='image/3.gif>";            //输出图像 3
			break;                                    //跳出语句
			case "4";                                 //如果值为 4
			echo "<img src='image/4.gif>";            //输出图像 4
			break;                                    //跳出循环
		}
	}
?>
```

（2）将该文件存储于\MR\03\046 文件夹下，并命名为 index.php。运行结果如图 3.19



所示。

技术要点



Note

break 关键字可以终止当前的循环，包括 while、do…while、for、foreach 和 switch 在内的所有控制语句。其在 for 语句中应用的示例如下：

```
for($i=1;$i<=4;$i++){                    //应用 for 循环控制语句输出表情头像
  if($i==4){                             //判断变量是否等于 4
        break;                           //如果等于 4，则使用 break 语句跳转循环
  }
```

本实例将 break 语句与 switch 语句联合使用，并控制页面中表情图的输出。

实例 047　控制页面中数据的输出数量

（实例位置：配套资源\SL\03\047　视频位置：配套资源\SP\03\047）

实例说明

break 语句可以指定跳出循环的层数，一般用于控制条件方面。本实例通过 break 语句实现控制页面中数据的输出数量，运行结果如图 3.20 所示。

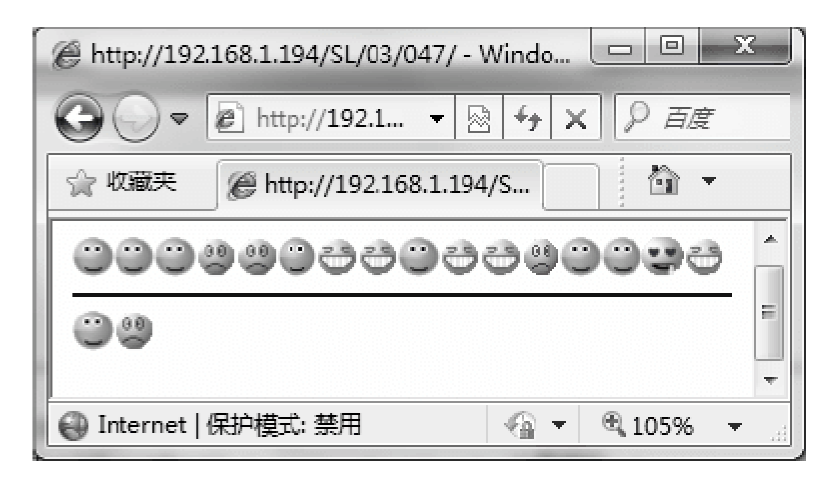


图 3.20　控制页面中数据的输出数量

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。首先，利用多重 for 循环嵌套输出全部图片。然后，在多重嵌套中定义 if 条件语句。当变量$rand 中的随机数值为 2 时，跳出 3 重循环；否则，正常输出图片。其代码如下：

```
<?php
    for($a = 0;$a < 2;$a++){                              //1 层循环
        for($b = 0;$b < 2;$b++){                          //2 层循环
            for($c = 0;$c < 4;$c++){                      //3 层循环
                $rand = rand(1,4);                        //1~4 随机数字
                echo "<img src='image/$rand.gif'>";       //输出图像
            }
        }
    }
    echo "<hr style='color:blue;'>";
```





```
        for($a = 0;$a < 2;$a++){                          //1 层循环
            for($b = 0;$b < 2;$b++){                      //2 层循环
                for($c = 0;$c < 4;$c++){                  //3 层循环
                    $rand = rand(1,4);                    //1~4 随机数字
                    if($rand != 2){                       //条件语句
                        echo "<img src='image/$rand.gif'>";   //输出图像
                    }else{
                        break 3;                          //结束 3 个层级的循环
                    }
                }
            }
        }
    ?>
```

（2）将该文件存储于\MR\03\047 文件夹下，并命名为 index.php。运行结果如图 3.20 所示。

技术要点

本实例运用 break 语句跳出指定的几重循环，其语法格式如下：

```
break $num;
```

参数$num 指定要跳出几层循环。break 关键字的控制流程如图 3.21 所示。

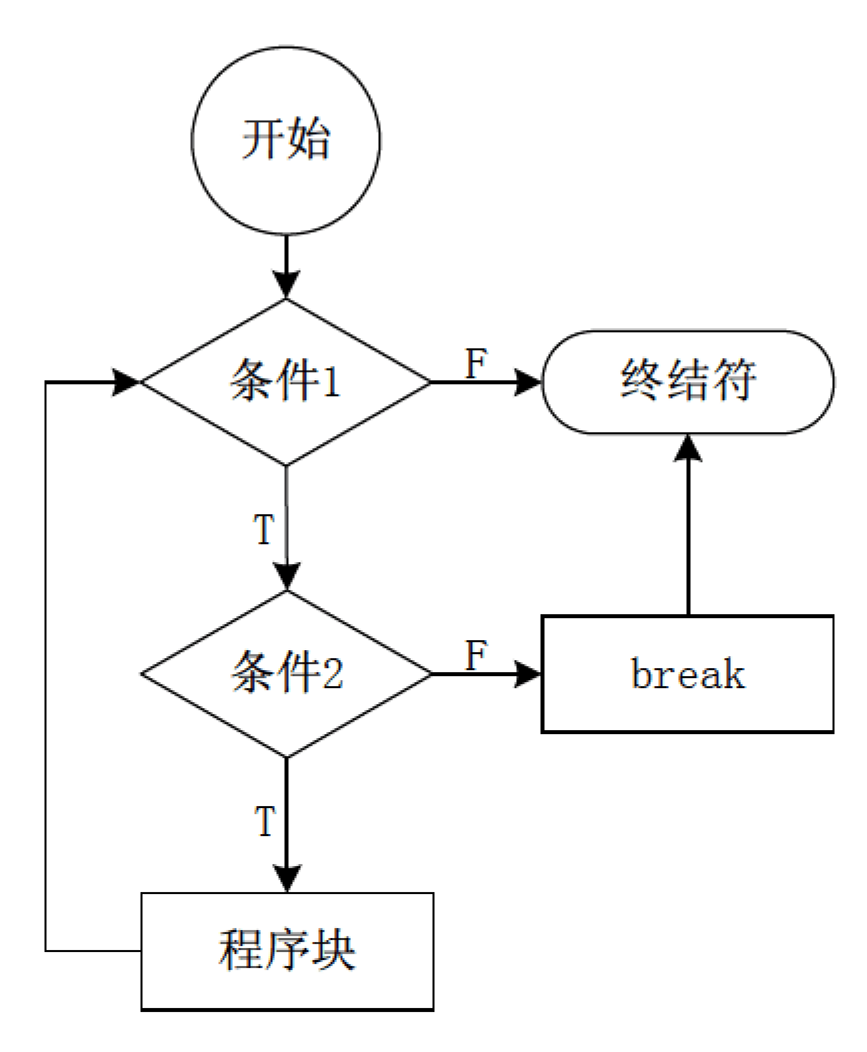


图 3.21　break 流程控制图

实例 048　考试成绩评定标准

（实例位置：配套资源\SL\03\048　视频位置：配套资源\SP\03\048）

实例说明

本实例通过 if 条件语句对文本框输入的考试成绩进行评定和选择，运行结果如图 3.22 所示。





Note

请输入考试成绩：100 评定

成绩优秀

图 3.22　考试成绩评定标准

实现过程

创建 index.php 文件。当单击“评定”按钮时，程序自动利用 POST 方法接收数据，并利用 if 语句对接收的数据进行判断，再用 echo 语句输出结果。其代码如下：

```
<?php
    if($_POST[sub]){                                          //通过 POST 方式传递参数
        if($_POST[text] == 100){                              //当成绩为 100 时
            echo "成绩优秀";                                  //输出
        }
        if($_POST[text] >= 60 and $_POST[text] < 100){        //当成绩大于等于 60 小于 100 时
            echo "成绩良好";                                  //输出
        }
        if($_POST[text] < 60){                                //当成绩小于 60 时
            echo "不及格";                                    //输出
        }
    }
?>
```

技术要点

if 语句是最简单的条件判定语句，它对某段程序的执行附加一个条件。如果条件成立，就执行这段程序；否则就跳过这段程序，去执行下面的程序。

本实例的关键点是 if 语句的灵活运用。当 expr 的值为 true 时，执行 statement1 语句。

```
if(expr){statement1;}
```

参数说明：

☑　expr：条件判断语句。

☑　statement1：符合条件的执行代码。



Web 技术

本章读者可以学到如下实例：

- 实例 049　通过客户端 IP 限制投票次数
- 实例 050　设计论坛登录界面
- 实例 051　可以上传图片的表单
- 实例 052　以文本域的形式显示数据信息
- 实例 053　验证用户注册信息是否合理
- 实例 054　省市级联动菜单
- 实例 055　省市县级联动菜单
- 实例 056　实现复选框中的全选、反选和不选
- 实例 057　上传图片预览
- 实例 058　通过下拉列表选择头像
- 实例 059　日期选择器
- 实例 060　在页面右下角弹出渐显的广告窗口
- 实例 061　树形导航菜单
- 实例 062　收缩式导航菜单
- 实例 063　控制登录用户的过期时间
- 实例 064　单点登录
- 实例 065　统计用户在线时间
- 实例 066　限制用户访问网站的时间
- 实例 067　SESSION 更换聊天室界面
- 实例 068　掌控登录用户的权限
- 实例 069　屏蔽页面刷新对计数器的影响
- 实例 070　SESSION 购物车
- 实例 071　清理 SESSION 缓存提高网站访问的效率
- 实例 072　限制上传文件的大小
- 实例 073　限制上传文件的类型
- 实例 074　通过链接方式下载
- 实例 075　上传多个文件到服务器
- 实例 076　通过 header()函数进行下载
- 实例 077　重新定义上传文件的名称





实例 049　通过客户端 IP 限制投票次数

（实例位置：配套资源\SL\04\049）

实例说明

本实例将制作一个简单的投票系统，通过获取客户端 IP 地址来限制用户的投票次数，每个 IP 只可以投票一次，如果重复投票则给出提示信息，运行结果如图 4.1 所示。







图 4.1　在线投票　　图 4.2　重复投票将弹出提示信息

实现过程

获取客户端 IP 地址需要使用$_SERVER[]全局数组中的$_SERVER['REMOTE_ADDR']参数。本实例将客户端 IP 地址存放到数据库中，当用户提交投票时，使用 if...else 语句判断该 IP 在数据库中是否存在，从而实现通过客户端 IP 地址限制投票次数。其具体代码如下：

```
<?php
//连接数据库
$conn=mysql_connect("localhost","root","111");
mysql_select_db("db_database04",$conn);
mysql_query("set names utf8");
$ip=$_SERVER['REMOTE_ADDR'];                                    //获取客户端 IP 地址
$insert="insert into tb_vote(IP)values('$ip')";    //定义添加数据语句
$select="select * from tb_vote where ip='$ip'"; //定义查询语句，查询 IP 地址在数据库中是否存在
if(isset($_POST['Submit']) and $_POST['Submit']=="提交投票"){      //判断按钮的执行操作
  $value=mysql_query($select,$conn);                               //执行查询操作
  if(mysql_num_rows($value)==0){                                   //判断查询结果是否等于 0
      $result=mysql_query($insert,$conn);//如果等于 0，执行添加语句，将 IP 地址添加到数据库中
      if($result){                                                 //判断添加操作是否成功执行
echo "<script>alert('投票成功！');window.location.href='index.php';</script>";    //如果成功执行，
则提示投票成功
      }else{
      echo "<script>alert('投票失败！');window.location.href='index.php';</script>";//否则提示投票
失败
      }
  }else{
      echo "<script>alert('您已经投过票了！');window.location.href='index.php';</script>";
  }
}
?>
```





技术要点

本实例主要使用$_SERVER[]全局数组中的$_SERVER['REMOTE_ADDR']参数来获取客户端 IP 地址。$_SERVER[ ]全局数组包含由 Web 服务器创建的信息，应用该数组可以获取服务器和客户端配置及当前请求的有关信息。下面对$_SERVER[ ]数组进行介绍，如表 4.1 所示。

表 4.1　$_SERVER[ ]全局数组

数 组 元 素	说　明
$_SERVER['SERVER_ADDR']	当前运行脚本所在服务器的 IP 地址
$_SERVER['SERVER_NAME']	当前运行脚本所在服务器主机的名称。如果该脚本运行在一个虚拟主机上，该名称则由那个虚拟主机所设置的值决定
$_SERVER['REQUEST_METHOD']	访问页面时的请求方法，例如，“GET”、“HEAD”、“POST”、“PUT”。如果请求的方式是 HEAD，PHP 脚本将在送出头信息后中止（这意味着在产生任何输出后，不再有输出缓冲）
$_SERVER['REMOTE_ADDR']	正在浏览当前页面用户的 IP 地址
$_SERVER['REMOTE_HOST']	正在浏览当前页面用户的主机名。反向域名解析基于该用户的 REMOTE_ADDR
$_SERVER['REMOTE_PORT']	用户连接到服务器时所使用的端口
$_SERVER['SCRIPT_FILENAME']	当前执行脚本的绝对路径名。注意：如果脚本在 CLI 中被执行，作为相对路径，例如 file.php 或者../file.php，$_SERVER['SCRIPT_FILENAME']将包含用户指定的相对路径
$_SERVER['SERVER_PORT']	服务器所使用的端口，默认为“80”。如果使用 SSL 安全连接，则这个值为用户设置的 HTTP 端口
$_SERVER['SERVER_SIGNATURE']	包含服务器版本和虚拟主机名的字符串
$_SERVER['DOCUMENT_ROOT']	当前运行脚本所在的文档根目录，在服务器配置文件中定义

实例 050　设计论坛登录界面

（实例位置：配套资源\SL\04\050　视频位置：配套资源\SP\04\050）

实例说明

论坛是许多网站不可或缺的模块之一，它提供一个空间，使人们相互之间能够进行各种沟通和交流。设计论坛模块的第一个步骤就是设计论坛登录界面，让用户根据自己注册的用户名和密码登录论坛。在本实例中，笔者设计了一个简单的论坛登录界面，效果如图 4.3 所示。



图 4.3　论坛登录界面

实现过程

具体步骤如下：

（1）利用 Dreamweaver 开发工具创建一个动态 PHP 页，并将其保存为“index.php”。



（2）添加一个表单，在表单中插入一个 3 行 2 列的表格。调整表格各行的高度和列的宽度。

（3）设置表格的 align 属性为“center”，使表格在表单中始终居中显示。

（4）向表格单元格中输入用户名称和密码信息，并添加文本字段和按钮。

（5）设置“密码”文本字段的类型（type）属性为“password”，使其以掩码的形式隐藏真实密码。

其主要代码如下：

```
<form action="" method="post">
    <table align="center">
        <tr><td>用户名</td><td><input type="text" name="te" size="20" /></td></tr>
        <tr><td>密   码</td><td><input type="password" name="tex" size="22" /></td></tr>
        <tr><td align="left"><input class="two"type="submit" name="sub" value="       " /></td>
        <td align="left"><input class="three" type="reset" name="res" value="       " /></td></tr>
    </table>
</form>
```

技术要点

在设计论坛登录界面时，主要采用表格标签进行界面布局。本实例设计了一个 3 行 2 列的表格，如图 4.4 所示。

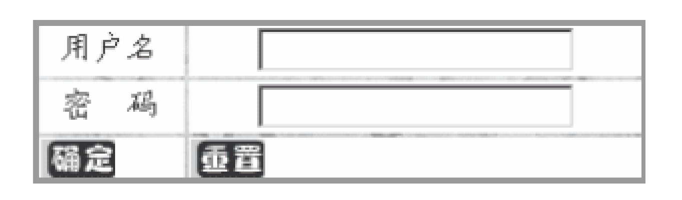


图 4.4 利用表格标签布局

这样，一个论坛登录界面的轮廓就形成了。接下来还需要为表格设置背景图像。利用表格标签的 background 属性就可以为表格添加背景图像了，代码如下：

```
<table width="392px" height="249px" align="center"><tr><td background="pic/tablebk.jpg">
```

实例 051 可以上传图片的表单

（实例位置：配套资源\SL\04\051）

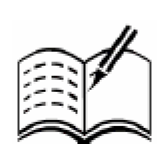
实例说明

许多大型网站为了吸引用户，会分配一些空间供用户使用。例如，大家经常使用的电子邮箱、QQ 个人空间等。为了能够让用户在个人空间中放置自己的内容，网站都会提供文件、图片等信息的上传功能。例如，在 QQ 个人空间中，用户可以将自己的照片上传上去，供好友欣赏。本实例中，我们利用文件域实现图片的上传功能，效果如图 4.5 所示。



图 4.5 可以上传图片的表单





实现过程

具体步骤如下：

（1）创建动态 PHP 文件，并命名为 index.php。

（2）添加一个表单，将 type 属性设置为 file。其代码如下：

```
<form action="" method="post" enctype="multipart/form-data">
        <input type="file" name="upfile" size="20" />
        <input  class="two"type="submit"  name="sub"  value="     
  " />
  </form>
```

（3）利用<table>标签编写网页框架。

（4）利用编码实现上传功能。其核心代码如下：

```
<?php
  if($_POST[sub]){                                          //单击上传按钮
        if($_FILES['upfile']['name'] == ''){                //判断上传文件名称是否为空
              echo "<script>alert('上传内容为空');</script>";
        }else{
              $info = $_FILES['upfile'];                    //将上传文件信息，保存在数组中
              if($info['size'] > 0 && $info['size'] < 1024 * 8000){ //判断上传文件的大小
                    $dir = 'upfiles/';                      //定义变量
                    $name = $info['name'];                  //设定文件名称
                    $rand = rand(0,10000000);
                    $name = $rand.date('YmdHis').$name;
                    $path = 'upfiles/'.$name;
                    if(!is_dir($dir)){          //判断文件夹是否存在，如果不存在则创建一个文件夹
                          mkdir($dir);
                    }
                    $move = move_uploaded_file($info['tmp_name'],$path);  //将上传文件移动到指
定位置
                    if($move == true){
                          echo "<script>alert('上传文件成功');</script>";
                    }
              }else{
                    echo "<script>alert('上传文件过大');</script>";
              }
        }
  }
?>
```

技术要点

在本实例中，实现照片上传的主要方式是利用文件域让用户加载本地文件，然后应用 PHP 函数 move_uploaded_file 实现文件的上传，效果如图 4.6 所示。



图 4.6　文本域加载本地文件

move_uploaded_file()函数：将上传的文件移动到新





位置。

```
bool move_uploaded_file ( string filename, string destination );
```

参数说明：

☑ string filename：指定的文件是合法的上传文件。

☑ string destination：定义上传文件的名称。

实例 052　以文本域的形式显示数据信息

（实例位置：配套资源\SL\04\052）

实例说明

在开发网站的过程中，一般涉及到用户注册的网站模块中都有一个用户注册的服务条款。个别的服务条款以整个网页进行显示，但是有的网站用户注册的服务条款篇幅过长，比较浪费网络空间，就不适合使用整个页面来显示了，最好的方式就是以文本域的形式来显示用户注册的服务条款，既节省页面，又美观大方。运行本实例，在文本域中显示用户注册的服务条款，效果如图 4.7 所示。

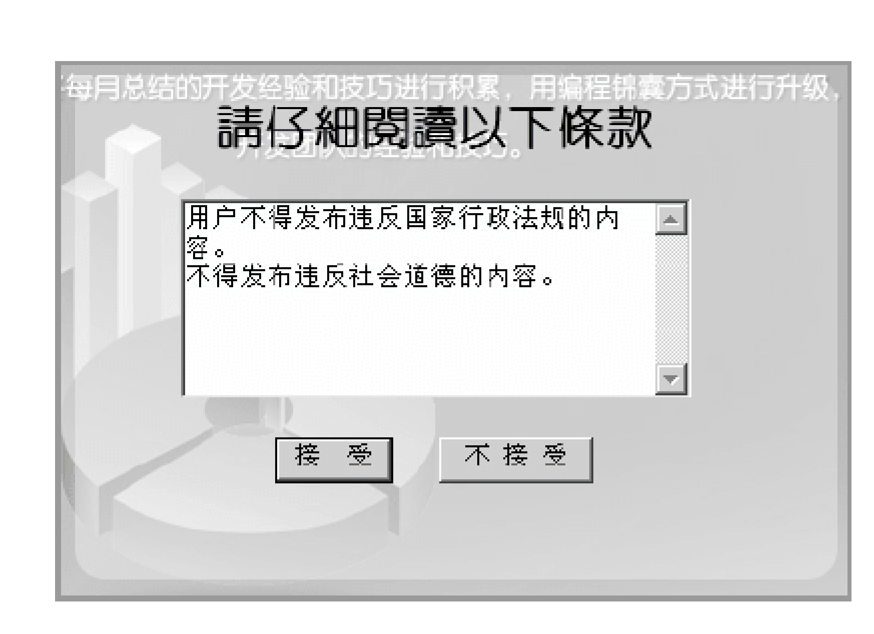


图 4.7　以文本域形式显示数据信息

实现过程

具体步骤如下：

（1）创建 index.php 文件，编写文本域代码，并通过 PHP 变量传递初始值。其代码如下：

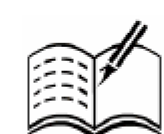
```
<?php
 $str = "用户不得发布违反国家行政法规的内容。                                //定义变量
不得发布违反社会道德的内容。";
?>
<form action="" method="post">
  <div align="center">
     <textarea name="txt" id="one" cols="30" rows="6"><?php echo $str;?></textarea>
     <br>
     <br>
        <input type="submit" name="agreen" id="agreen" value="接　受" />
             
        <input type="button" name="disagreen" id="disagreen" value="不 接 受" />
    </div>
</form>
```

技术要点

本实例应用表单中的文本域显示信息，主要就是设置文本域的初始值。代码如下：

```
<textarea name="test" cols="45" rows="8" id="test"><?php echo $myrow[text];?></textarea>
```





参数说明：

☑ name：文本域的名称。

☑ cols：文本域的字符宽度。

☑ rows：文本域显示的行数。

实例 053 验证用户注册信息是否合理

（实例位置：配套资源\SL\04\053）

实例说明

本实例主要实现判断用户输入的注册信息是否合理，当用户输入的注册信息不合理时给出相应的提示，运行结果如图 4.8 所示。



图 4.8 验证用户注册信息是否合理

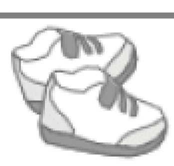
实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计用户注册信息页面，效果如图 4.8 所示。

（2）创建 javascript 脚本文件，完成验证用户注册信息是否合理，代码如下：

```
<script language="javascript">
function checkemail(email){
  var str=email;
   //在 JavaScript 中，正则表达式只能使用"/"开头和结束，不能使用双引号
  var Expression=/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/;     //定义 E-mail 地址格式
  var objExp=new RegExp(Expression);
  if(objExp.test(str)==true){
         return true;
  }else{
```




Note

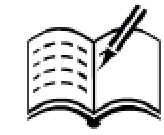
```
            return false;
        }
    }
    function checktel(tel){
      var str=tel;
       //在 JavaScript 中，正则表达式只能使用"/"开头和结束，不能使用双引号
      var Expression=/(\d{3}-)(\d{8})$|(\d{4}-)(\d{7})$/;                    //定义电话格式
      var objExp=new RegExp(Expression);
      if(objExp.test(str)==true){
            return true;
      }else{
            return false;
      }
    }
    function checkit(){                                        //自定义函数
      if(form1.name.value==""){                                //判断用户名是否为空
                  alert("请输入用户名!");
                        form1.name.select();
                    return false;
                  }
      if(form1.pwd.value==""){                                 //判断密码是否为空
                  alert("请输入密码!");
                        form1.pwd.select();
                    return false;
                  }
      if(form1.qpwd.value==""){                                //判断确认密码是否为空
                  alert("请输入确认密码!");
                        form1.qpwd.select();
                    return false;
                  }
      if(form1.pwd.value!=form1.qpwd.value){                   //判断两次输入的密码是否一致
                    alert("两次输入的密码不一致!");
                    form1.pwd.select();
                    return false ;
               }
      if(form1.email.value==""){                               //判断 E-mail 是否为空
                  alert("请输入 email!");
                        form1.email.select();
                    return false;
                  }
      if(!checkemail(form1.email.value)){                      //判断 E-mail 地址格式是否正确
                    alert("您输入 Email 地址不正确!");
                    form1.email.select();
                    return false;
      }
      if(form1.tel.value==""){                                 //判断电话是否为空
                  alert("请输入电话!");
                        form1.tel.select();
                    return false;
```



```
        }
    if(!checktel(form1.tel.value)){                //判断电话号码格式是否正确
            alert("您输入的电话不正确!");
            form1.tel.select();
            return false;
    }

        return true;

    }
</script>
```



技术要点

本实例主要通过 JavaScript 脚本技术来实现判断用户输入的注册信息是否合理。

实例 054　省市级联动菜单

（实例位置：配套资源\SL\04\054　视频位置：配套资源\SP\04\054）

实例说明

在进行网站开发的过程中，经常会用到相互关联的菜单。例如，省、市、县三级联动或者省市两级联动菜单。在本实例中，笔者讲解如何实现省市两级联动菜单，运行效果如图 4.9 所示。



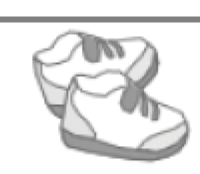
图 4.9　省市级联动菜单

实现过程

具体步骤如下：

（1）创建脚本文件，引入 CSS 样式表和 JavaScript 脚本。编写一级下拉列表，设置 <option>标签属性“value”的值。其代码如下：

```
<select id="select1">
    <option value="" selected>请选择</option>
    <option value="1">吉林</option>
```




```
            <option value="2">辽宁</option>
        </select>
```



（2）编写 JavaScript 代码，当选则一级下拉列项时，触发 change 事件。代码如技术要点中所示。

技术要点

本实例通过设置 change 事件实现联动的下拉列表，并且通过一级下拉列表传递的参数不同，显示不同的两级下拉列表信息。实现这些功能的 JavaScript 代码如下：

```
        $("#select1").change( function() {
            var id = $("#select1").val();
            if(id == 1){
                $.get('index.php', null, function(data){
                    $("span").empty();
                    $("span").append("<select><option>长春</opton><option>松原</opton><option>
通辽</opton></select>");
                });
            }else{
                $.get('index.php', null, function(data){
                    $("span").empty();
                    $("span").append("<select><option>大连</opton><option>旅顺</opton><option>
鲅鱼圈</opton></select>");
                });
            }
        });
```

实例 055　省市县级联动菜单

（实例位置：配套资源\SL\04\055）

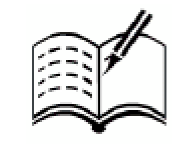
实例说明

在实例 054 中，笔者已经讲解如何实现两级联动菜单，下面笔者向用户讲解如何实现省、市、县三级联动菜单。运行本实例，效果如图 4.10 所示。



图 4.10　省市县级联动菜单





实现过程

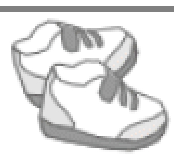
具体步骤如下：

（1）创建脚本文件，并命名为 index.php。然后创建 form 表单，并编写下拉列表框，定义两级联动和三级联动的下拉列表选项，代码如下：

```
<select id="select1">
        <option value="" selected>请选择</option>
        <option value="1">吉林</option>
        <option value="2">辽宁</option>
  </select>
  <span id='two'></span>
  <span id='three'></span>
```

（2）创建 JavaScript 脚本文件，当触发 change 事件时，实现联动效果，代码如下：

```
$("#select1").change( function() {
        var id = $("#select1").val();
        if(id == 1){
                $.get('index.php', null, function(data){
                        $("#two").empty();
                        $("#two").append("<select id='select2'><option>请选择</opton><option value='1'>
长春</opton><option value='2'>松原</opton></select>");
                        $("#select2").change( function() {
                                var page = $("#select2").val();
                                if(page == 1){
                                        $("#three").empty();
                                        $("#three").append("<select><option> 双 阳 </opton><option> 农 安
</opton></select>");
                                }else{
                                        if(page == 2){
                                                $("#three").empty();
                                                $("#three").append("<select><option> 乾 安 </opton><option> 大
安</opton></select>");
                                        }
                                }
                        });
                });
        }else{
                $.get('index.php', null, function(data){
                        $("#two").empty();
                        $("#two").append("<select id='select3'><option value='1'> 大 连 </opton><option
value='2'>旅顺</opton></select>");
                        $("#select3").change( function() {
                                var page = $("#select3").val();
                                if(page == 1){
                                        $("#three").empty();
                                        $("#three").append("<select><option> 西 岗 </opton><option> 沙 河 口
</opton></select>");
                                }else{
```

```
                                if(page == 2){
                                        $("#three").empty();
                                        $("#three").append("<select><option> 金 州 </opton><option> 瓦
房店</opton></select>");
                                }
                        }
                });
        });
    }
});
```

技术要点

本实例与实现两级联动的菜单思路大致相同，只不过套用多层的 if...else 语句，来实现省、市、县三级联动菜单。

实例 056　实现复选框中的全选、反选和不选

（实例位置：配套资源\SL\04\056　视频位置：配套资源\SP\04\056）

实例说明

复选框应用在用户注册系统中，如个人爱好选择等。本实例所讲述的三个操作：全选、反选、不选，只是根据实际情况，对复选框实现的快捷操作。运行本实例，全选的效果如图 4.11 所示。



图 4.11　实现全选

实现过程

具体步骤如下：

（1）创建 js 文件夹，编写 reg.js 脚本文件。在 reg.js 中，编写自定义函数，实现全选、反选和不选功能。reg.js 文件的关键代码如下：

```
function uncheckAll(form1,status) {                          //不选
    var elements = form1.getElementsByTagName('input');      //获取 input 标签
```

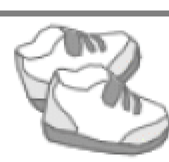





```
    for(var i=0; i<elements.length; i++){            //根据标签的长度执行循环
        if(elements[i].type == 'checkbox') {         //判断对象中元素的类型
            if(elements[i].checked==true){           //判断当 checked 的值为 true 时
                elements[i].checked=false;           //为 checked 赋值为 false
            }
        }
    }
}
function checkAll(form1,status){                     //全选
    var elements = form1.getElementsByTagName('input');
    for(var i=0; i<elements.length; i++){
        if(elements[i].type == 'checkbox') {
            if(elements[i].checked==false){
                elements[i].checked=true;
            }
        }
    }
}
function switchAll(form1,status) {                   //反选
    var elements = form1.getElementsByTagName('input');
    for(var i=0; i<elements.length; i++){
        if(elements[i].type == 'checkbox'){
            if(elements[i].checked==true){
                elements[i].checked=false;
            }else if(elements[i].checked==false){
                elements[i].checked=true;
            }
        }
    }
}
```

（2）创建 index.php 页面，输出会员信息，并添加图像按钮。通过 onclick 事件调用 JavaScript 自定义函数，实现全选、反选、不选和删除的功能。其关键代码如下：

```
<script language="javascript" src="js/reg.js"></script>
<form method="post" name="form1" id="form1" action="index_ok.php">
<tr>
<td width="62" align="center"><input name="conn_id[]" type="checkbox" id="conn_id[]" value="1"
/></td>
    <td width="242">欧阳</td>
    <td width="243">PHP</td>
    <td width="485">部门经理</td>
    <td width="485">29</td>
</tr>
<tr>
<td colspan="5" align="center"><!--通过 onclick 事件调用自定义函数，执行相应的操作-->
<img src="images/bg_19-20.jpg" onclick="checkAll(form1,status)" width="62" height="25" />
  <img src="images/bg_14-14.jpg" onclick="switchAll(form1,status)"   width="62" height="25" />
  <img src="images/bg_07-08.jpg" width="62" height="25" onclick="uncheckAll(form1,status)" />
```




```
<input type="image" name="imageField" src="images/bg_14.jpg" />
</td>
</tr>
</form>
```



Note

技术要点

复选框的全选、反选和不选主要通过 JavaScript 脚本的自定义函数来完成。

（1）在通过 JavaScript 脚本中的自定义函数完成复选框的全选、反选和不选功能中，应用的是 getElementsByTagName，获取指定标签的名称，其返回值是一个包含标签信息的 object。

（2）根据 getElementsByTagName 标签返回的对象，判断标签类型（type）的值是否为 checkbox。

（3）当标签类型 type 的值为 checkbox 时，为标签中的 checked 赋值。当 checked 的值为 true 时，为它赋值为 false；当 checked 的值为 false 时，为它赋值为 true。

（4）在页面中，通过 script 标签调用 js 文件夹下的文件，应用 onclick 事件调用自定义函数，完成全选、反选和不选功能的实现。

实例 057　上传图片预览

（实例位置：配套资源\SL\04\057）

实例说明

上传图片时最好为程序设置预览功能，防止用户意外传错图片。还有一个很重要的模块就是获取图片的地址，方便用户网站的推广。运行本实例，完成图片预览和图片地址获取的功能，运行结果如图 4.12 和图 4.13 所示。



图 4.12　上传图片预览



图 4.13　获取图片地址

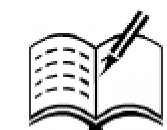
实现过程

（1）创建 index.php 文件，引入 CSS 样式和 in.js 文件以及 jQuery 库文件 jquery-1.3.2.js。创建 form 表单，设置文件域属性，添加预览图片、图片地址和清除的按钮。其代码如下：

```
<table width="600" height="450" align="center" background="pic/bg.jpg">
  <tr>
```







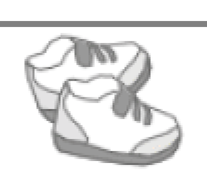

```
        <td align="center"><div class="n"></div>
    <table align="center">
      <tr>
    <td>
            <span class="o">
            <span class="e"><input class="one" id="one" type="file"></span>
            <input class="one" id="two" type=button value="预览图片">
            <input class="one" id="three" type=button value="图片地址">
            <input class="one" id="four" type=button value="清除" >
            </span>
    </td>
      </tr>
    </table>
     </td>
      </tr>
    </table>
```

（2）创建 JavaScript 脚本文件 in.js，代码如技术要点中所示。

技术要点

本实例通过 JavaScript 的 click 事件，实现图片的预览、获取地址和清除上传文本框信息的功能。其核心代码如下：、

```
$(document).ready(function(){
  $("#two").click(function(){                                   //单击预览图片
        var value = $("#one").val();                            //取得上传地址
        if(value == ""){                                        //判断地址是否为空
              alert("地址为空");
        }else{
              $(".n").append("<img id='five' src="+value+">");  //显示图片
              $("#two").click(function(){
                    $(".n").empty();
                    $(".n").append("<img id='five' src="+value+">");
              });
        }
  });
  $("#three").click(function(){                                 //显示图片地址
        var value = $("#one").val();
        if(value == ""){
              alert("地址为空");
        }else{
              alert(value);
        }
  });
  $("#four").click(function(){                                  //清除文本框信息
        $(".e").empty();
        $(".e").append("<input class='one' id='one' type='file'>");
  });
});
```





实例 058　通过下拉列表选择头像

（实例位置：配套资源\SL\04\058）

实例说明

在腾讯的 QQ 系统中，我们可以通过单击头像而实现更换新的头像。其实这个功能相对来说是比较简单的。运行本实例，通过下拉列表实现头像的选择，效果如图 4.14 所示。



图 4.14　通过下拉列表选择头像

实现过程

具体步骤如下：

（1）创建 index.php 文件，引入 CSS 样式和 JavaScript 脚本文件 in.js 以及 jQuery 库，编写下拉列表和标签，具体代码如下：

```
<table align="center">
  <tr>
    <td>
      <select name="select">
            <option value="">请选择性别</option>
            <option value="nan">男性</option>
            <option value="nv">女性</option>
      </select>
      <div id="o"></div><div id="n"></div>
 </td>
  </tr>
</table>
```

（2）创建 JavaScript 脚本 in.js 文件，代码如技术要点中所示。

技术要点

本实例通过 JavaScript 的 change 事件和实例“省市级联动菜单”中的两级联动菜单实现。核心代码如下：

```
$(document).ready(function(){
  $("select").change(function(){
```





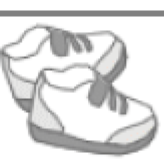
```
                var value = $(this).val();
                if(value == ""){
                    alert("请选择性别");
                }else{
                    if(value == "nan"){
                        $("#o").empty();
                        $("#o").append("<select id='one'><option value='pic/1.jpg'>头像 1</option><option value='pic/2.jpg'>头像 2</option><option value='pic/3.jpg'>头像 3</option><option value='pic/4.jpg'>头像 4</option><option value='pic/9.jpg'>头像 5</option></select>");
                        $("#n").empty();
                        $("#n").append("<img src='pic/1.jpg'>");
                        $("#one").change(function(){
                            var va = $(this).val();
                            $("#n").empty();
                            $("#n").append("<img src="+va+">");
                        });
                    }else{
                        $("#o").empty();
                        $("#o").append("<select id='one'><option value='pic/5.jpg'>头像 1</option><option value='pic/6.jpg'>头像 2</option><option value='pic/7.jpg'>头像 3</option><option value='pic/8.jpg'>头像 4</option><option value='pic/10.jpg'>头像 5</option></select>");
                        $("#n").empty();
                        $("#n").append("<img src='pic/5.jpg'>");
                        $("#one").change(function(){
                            var va = $(this).val();
                            $("#n").empty();
                            $("#n").append("<img src="+va+">");
                        });
                    }
                }
            });
        });
```

实例 059　日期选择器

（实例位置：配套资源\SL\04\059）

实例说明

基于目前日期型数据格式有多种，采用录入方式相对来说比较烦琐，而且采用这种方式也不利于日期格式的统一，所以可以在信息录入页面中加入一个简单的日期选择器来解决上述问题。运行本实例，在新奥家电连锁后台管理系统的销售查询页面中，单击“售货日期”文本框后的日期选择图标，会弹出“日期选择器”对话框，选择售货日期的起始日期，单击“确定”按扭，即可成功地将选择的日期添加到对应的日期文本框中，运行结果如图 4.15 所示。




Note



图 4.15 日期选择器

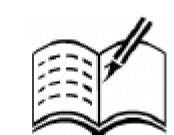
实现过程

具体步骤如下：

（1）在父窗口中创建函数 open_day_from()和函数 open_day_to()，实现子窗口的弹出和父窗口与子窗口之间的信息传递，代码如下：

```
<table width="250" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
 <form name="form2">
  <tr bgcolor="#826650">
    <td width="13"><div align="center"><a href="#" onClick="addday()"></a></div></td>
    <td width="80">
 <select name="n1">
 <?php
  for($i=2005;$i<=2050;$i++) {
 ?>
      <option value=<?php echo $i;?>><?php echo $i;?></option>
 <?php
   }
 ?>
    </select>
    年</td>
    <td width="64"><select name="y1">
 <?php
  for($i=1;$i<=12;$i++) {
 ?>
      <option value=<?php echo $i;?>><?php echo $i;?></option>
   <?php
   }
 ?>
    </select>
    月</td>
    <td width="77"><select name="r1">
 <?php
  for($i=1;$i<=31;$i++) {
 ?>
      <option value=<?php echo $i;?>><?php echo $i;?></option>
  <?php
   }
```





```
        ?>
            </select>
            日</td>
            <td width="16"><div align="center"><a href="#" onClick="subday()"></a></div></td>
        </tr>
        </form>
    </table>
    <table width="250" height="20" border="0" align="center" cellpadding="0" cellspacing="0">
        <tr>
            <td bgcolor="#826650"><div align="center"><input type="button" onClick="close_day()" class=
"buttoncss" value="确定"></div></td>
        </tr>
    </table>
```

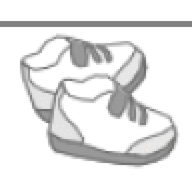
（2）单击“确定”按钮，调用 close_day()函数实现日期的传值，代码如下：

```
    <script language="javascript">
    function close_day(){
    window.returnValue=document.form2.n1.value+"-"+document.form2.y1.value+"-"+document.form2.r
1.value;
        window.close();
    }
    </script>
```

技术要点

由于日期为连续的数字，开发网页时将这些信息一个接一个地写在<option>与</option>标记之间固然可行，但这会降低程序开发时间。开发本实例时，笔者将在下拉列表框中显示的连续数据通过 PHP 的循环结构显示出来，这样更灵活方便。实现该过程的代码如下：

```
    <form name="form2">
        <tr bgcolor="#826650">
            <td width="13"><div align="center"><a href="#" onClick="addday()"></a></div></td>
            <td width="80">
        <select name="n1">
        <?php
         for($i=2005;$i<=2050;$i++){
        ?>
                <option value=<?php echo $i;?>><?php echo $i;?></option>
        <?php
            }
        ?>
            </select>
            年</td>
            <td width="64"><select name="y1">
        <?php
         for($i=1;$i<=12;$i++){
        ?>
                <option value=<?php echo $i;?>><?php echo $i;?></option>
```






```
      <?php
      }
    ?>
        </select>
        月</td>
        <td width="77"><select name="r1">
    <?php
     for($i=1;$i<=31;$i++){
    ?>
            <option value=<?php echo $i;?>><?php echo $i;?></option>
     <?php
      }
    ?>
        </select>
        日</td>
        <td width="16"><div align="center"><a href="#" onClick="subday()"></a></div></td>
     </tr>
     </form>
```

实例 060 在页面右下角弹出渐显的广告窗口

（实例位置：配套资源\SL\04\060）

实例说明

广告作为网站最大的盈利手段，任何网站都不可能将其省略。那么，如何才能使网站既可以从广告中获利，又能使用户流畅地阅读网站信息呢？本实例将介绍如何制作一个从首页右下角弹出的渐显广告窗体，在用户登录网站时自动从右下角渐渐弹出，然后由用户手动将其关闭。运行结果如图 4.16 所示。



图 4.16 右下角渐显广告

实现过程

本实例主要应用 window 对象的 open 方法调用打开一个在首页右下角弹出渐显的广告窗口，在网站首页中添加如下代码：

```
<script language="javascript">
var newformW=300;
var newformH=180;
```



```
function pp(){
  var T=window.screen.width-newformW;
  var L=window.screen.height+newformH;
 var  name=window.open("advertise.htm","","width="+newformW+",height="+newformH+",top="+T+",
left="+L);
}
pp();
</script>
```

技术要点

本实例主要应用 JavaScript 脚本的 window 对象，window 对象主要应用在 HTML 中打开窗口，应用极为普遍，但也有一些缺陷。用户浏览器决定窗口的外观，设计者左右不了其窗口的大小及样式，但 JavaScript 给了程序这种控制权。在 JavaScript 中，可以使用 window 对象来实现对窗口的控制。

实例 061　树形导航菜单

（实例位置：配套资源\SL\04\061　视频位置：配套资源\SP\04\061）

实例说明

树形结构能够以层次形式展示信息，用它来描述具有上下级关系的内容再恰当不过了。本实例利用菜单的树形结构来描述企业人事组织架构，效果如图 4.17 所示。

人事组织架构
程序开发部门
Visual C++
Visual Basic
办公软件部门
Word部门
Excel部门
客户服务部门

图 4.17　树形导航菜单

实现过程

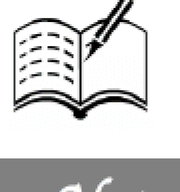
具体步骤如下：

（1）使用 Dreamweaver 创建一个 PHP 动态页，并保存为“index.php”文件。

（2）创建一个表单，在表单中添加一个 2 行 1 列的表格，在表格的第 2 行再添加一个 2 行 2 列的表格。表格的第一列用于显示节点前的图标，第 2 列描述节点文本。实例主要代码如下：

```
<form id="form1" name="form1" method="post" action="" >
  <table width="372"   border="1" align="center"   bgcolor="#E6F2F2" bordercolor="#478D8D"
  style="border-style:none" cellspacing="0" >
    <tr>
      <td     scope="col" align="center" style="border-bottom-style:none" >人事组织架构</td>
    </tr>
    <tr>
      <td >
           <table width="100%"   border="0" cellpadding="0" cellspacing="0" align="left">
        <?php                                              //嵌入 PHP 脚本
          $xml_file = simplexml_load_file("org.xml");      //从 XML 文件中加载节点
             foreach($xml_file->children() as $node){      //遍历根节点
```



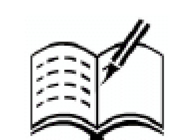



```
?>
        <tr>
         <td width="38%" align="right">                        //设置节点图标
              <img src="images/rootnode.gif" width="15" height="15" border="0" >     </img>
         </td>
           <td  width ="62%"  left="0px" align="left" ><a href="#" onclick= "ShowNode(<?php
echo $node ?>)">  
         <?php echo $node ?></a>                                //设置节点文本
           </td>
        </tr>
        <tr>
        <td width ="38%" align="center" >
        </td>
               <td   width ="62%" >
                    <div id= <?php echo $node ?> style = "display:none" class="HideDiv" >//定
义 DIV
                 <table width="100%" border="0" cellpadding="0" cellspacing="0" align="left" >
                    <?php
                      foreach($node->children() as $value){     //遍历子节点
                    ?>
                         <tr>
                        <td width ="10%"   align="right" >
                           <img   src="images/childnode.gif"  width="15"  height="15"  border=
"0" > </img>
                       </td>
                       <td   width ="90%"   align="left"   ><a href="#" >  <?php
echo $value?></a>
                       </td>
                    </tr>
                    <?php
                         }
                    ?>
                 </table>
              </div>
               </td>
        </tr>
      <?php
     }
    ?>
      </table>
    </td>
  </tr>
</table>
<script language="javascript">
function ShowNode(ParentID) {                                   //隐藏或显示子节点
      var Div = ParentID;
      if (Div.style.display == "none"){
             Div.style.display = "block";                       //显示 DIV，以此来显示子节点
      }else if (Div.style.display == "block"){
             Div.style.display = "none";                        //隐藏 DIV，间接隐藏子节点
```



```
            }
        }
        </script>
    </form>
```

（3）人事组织的架构存储于 XML 文件 org.xml 中。



技术要点

本实例中，实现菜单的树形显示是通过<table>结合 DIV 实现的。父节点利用表格来显示，子节点利用 DIV 嵌套表格来实现。这样做的目的是容易对子节点的显示和隐藏进行控制。例如，当用户单击父节点时，如果子节点没有展开，将 DIV 设置为可见就可以显示子节点。如果子节点可见，将 DIV 设置为不可见就可以隐藏子节点。

实例 062　收缩式导航菜单

（实例位置：配套资源\SL\04\062）

实例说明

在网站中不仅可以设置导航条，而且还可以设置导航菜单。由于菜单内容比较多，在同一页面中显示会比较杂乱，所以目前大多数的设计者都采用收缩式的导航菜单。运行本实例，当浏览者单击“网站管理”超链接时，在其下方将弹出导航菜单，如图 4.18 所示，浏览者再次单击“网站管理”超链接时，导航菜单又收缩回去，页面中不再显示菜单中的内容。





图 4.18　收缩式导航菜单

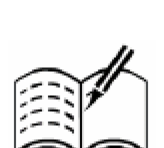
实现过程

具体步骤如下：

（1）显示菜单的自定义函数，代码如下：

```
<script language=javascript>
function show(obj,maxg,obj2){
    if(obj.style.pixelHeight<maxg) {
```






```
          obj.style.pixelHeight+=maxg/10;
      obj.filters.alpha.opacity+=20;
      obj2.background="images/title_hide.gif";
          if(obj.style.pixelHeight==maxg/10)
         obj.style.display='block';
      myObj=obj;
      mymaxg=maxg;
      myObj2=obj2;
      setTimeout('show(myObj,mymaxg,myObj2)','5');
       }
   }
   </script>
```

（2）隐藏菜单的自定义函数，代码如下：

```
   <script language=javascript>
   function hide(obj,maxg,obj2){
      if(obj.style.pixelHeight>0) {
          if(obj.style.pixelHeight==maxg/5)
         obj.style.display='none';
          obj.style.pixelHeight-=maxg/5;
      obj.filters.alpha.opacity-=10;
      obj2.background="images/title_show.gif";
      myObj=obj;
      mymaxg=maxg
      myObj2=obj2;
      setTimeout('hide(myObj,mymaxg,myObj2)','5');
       }
      else
          if(whichContinue)
         whichContinue.click();
   }
   </script>
```

（3）单击菜单上的文字超链接时，显示当前菜单，并隐藏前一个菜单，代码如下：

```
   <script language=javascript>
   function chang(obj,maxg,obj2){
      if(obj.style.pixelHeight) {
          hide(obj,maxg,obj2);
            nopen='';
            whichcontinue='';
       }
      else
          if(nopen){
               whichContinue=obj2;
            nopen.click();
      }
      else{
         show(obj,maxg,obj2);
         nopen=obj2;
```



```
            whichContinue='';
        }
    }
</script>
```

（4）在表格的相关鼠标事件中调用自定义的方法和属性来改变收缩菜单的显示和隐藏，代码如下：

```
<TD class=list_title id=list1 onmouseover="this.typename='list_title2';" onclick=chang(menu1,60,list1); onmouseout="this.typename='list_title';" background="images/title_hide.gif" height=25><SPAN>网站管理</SPAN> </TD>
```



技术要点

本实例主要是利用显示隐藏表格来实现收缩式导航菜单的功能。单击导航超链接，显示当前菜单的内容，隐藏上一个显示的菜单。在隐藏菜单时，让其有规律地隐藏，进而实现动画的效果。

实例 063　控制登录用户的过期时间

（实例位置：配套资源\SL\04\063）

实例说明

Cookie 是在 HTTP 协议下，将服务器传递给浏览器的少量数据保存到用户浏览器的一种方式。通过这种方式，即使在浏览器被关闭和连接中断的情况下，用户仍然可以维护状态数据。本实例在创建和读取 Cookie 的同时，设置 Cookie 的生命周期，实现控制登录用户的过期时间。运行结果如图 4.19 所示。





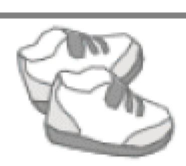
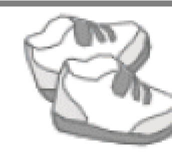
图 4.19　控制登录用户的过期时间

实现过程

具体步骤如下：

（1）创建 index.php 文件，并编写用户登录页面，将用户登录信息提交到 index_ok.php 文件。

（2）创建 index_ok.php 文件，获取表单提交的用户登录信息，并判断登录的用户名和密码是否正确。如果正确，则将用户名和密码赋给指定的 Cookie 变量，并设置 Cookie 的过期时间，然后跳转到 cookie.php 页面；否则，直接给出提示信息，并重新跳转到登录页。其关键代码如下：

Note

```
<?php
if($_POST['user']=="mr" && $_POST['pass']=="mrsoft"){            //判断用户名和密码是否正确
  setCookie("user",$_POST['user'],time()+60)or die("禁止 cookie");    //创建 Cookie
  setCookie("pass",$_POST['pass'],time()+60)or die("禁止 cookie");    //创建 Cookie
  echo "<script>alert('登录成功！'); window.location.href='cookie.php';</script>";
}else{
  echo "<script>alert('用户名或者密码不正确！'); window.location.href='index.php';</script>";
}
?>
```

（3）创建 cookie.php 文件，并判断 Cookie 变量的值是否存在。如果存在，则输出本页内容；否则，给出提示信息并跳转到登录页面。其关键代码如下：

```
<?php
  if($_COOKIE['user']=="mr" && $_Cookie['pass']=="mrsoft"){
        echo "欢迎光临！";
  }else{
        echo "<script>alert('Cookie 已经过期，请重新登录'); window.location.href='index.php';
</script>";
  }
?>
```

技术要点

（1）setcookie()函数：设置 Cookie。

```
bool setcookie ( string name [, string value [, int expire [, string path [, string domain [, int secure]]]]]);
```

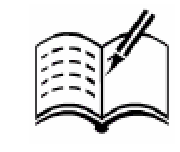
该函数的参数说明如表 4.2 所示。

表 4.2　setcookie()函数的参数说明

参　　数	说　　明	举　　例
name	Cookie 的变量名	可以通过$_COOKIE['cookiename ']调用变量名为 cookiename 的 Cookie
value	Cookie 变量的值，该值保存在客户端，不能用来保存敏感数据	可以通过$_COOKIE['values ']获取名为 values 的值
expire	Cookie 的过期时间，expire 是标准的 UNIX 时间标记，可以用 time()函数或 mktime()函数获取，单位为秒	如果不设置 Cookie 的过期时间，那么 Cookie 将永远有效，除非手动将其删除
path	Cookie 在服务器端的有效路径	如果该参数设置为“/”，则它就在整个 domain 内有效，如果设置为“/5.1”，它就在 domain 下的/5.1 目录及子目录内有效。默认是当前目录
domain	Cookie 有效的域名	如果要使 Cookie 在 mrbccd.cn 域名下的所有子域都有效，应该设置为 mrbccd.cn
secure	指明 Cookie 是否仅通过安全的 HTTPS，值为 0 或 1	如果值为 1，则 Cookie 只能在 HTTPS 连接上有效；如果值为默认值 0，则 Cookie 在 HTTP 和 HTTPS 连接上均有效

（2）$_COOKIE：经由 HTTP Cookies 方法提交至脚本的变量，用于获取 Cookie 变量





的值。

实例 064　单点登录

（实例位置：配套资源\SL\04\064）

实例说明

运行本实例，如果是第一次登录，那么将进入到登录页面，登录成功后将进入到博客主页，如图 4.20 所示。在主页面可以单击博客中的“我的论坛”超链接，以此用户的身份直接进入到论坛系统主页中，如图 4.21 所示。同样也可以单击论坛系统中的“博客”，跳转到博客的页面中。

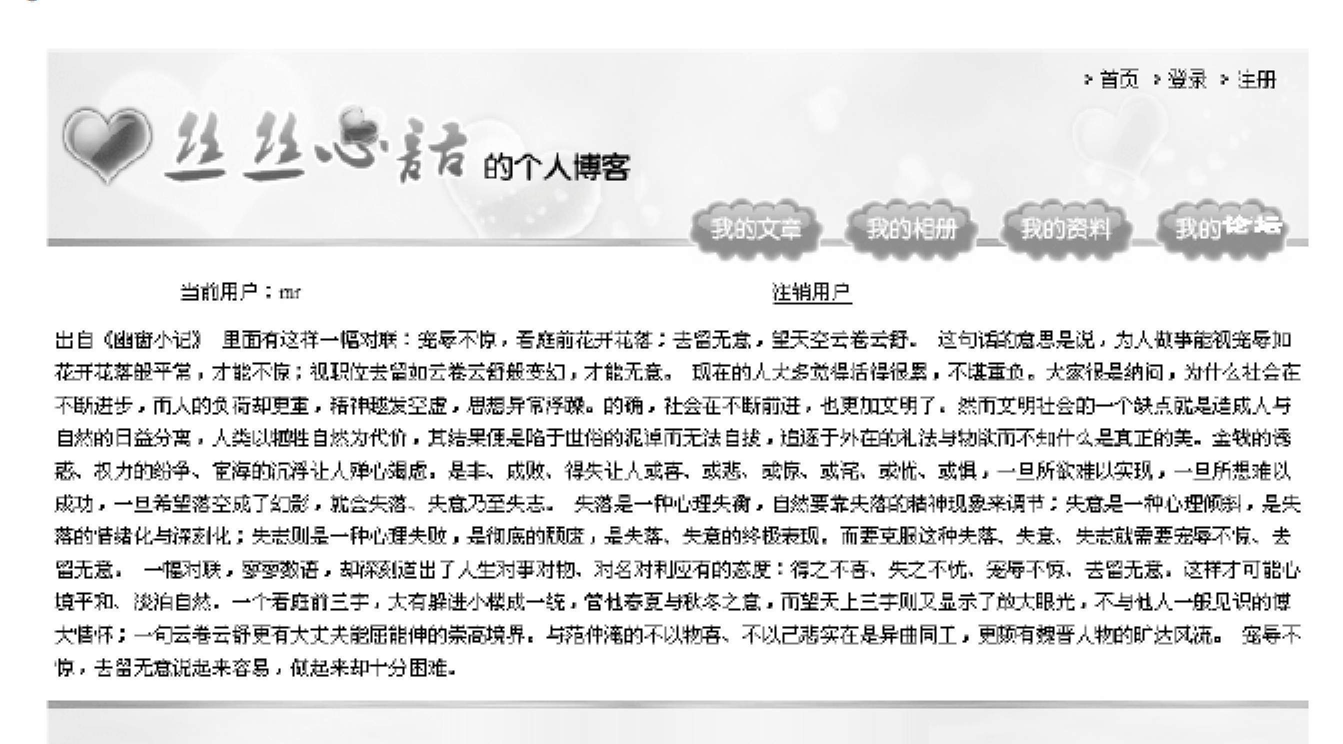


图 4.20　进入博客主页面

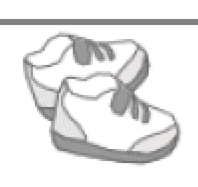


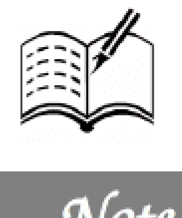
图 4.21　进入论坛系统主页

实现过程

这里以博客网站为例，讲解其具体的操作步骤：

（1）创建 index.php 页面，添加用户登录的表单，将登录信息提交到 index_ok.php 页面中。在 index.php 页面中首先连接数据库，然后判断 Cookie 的值是否为空，如果 Cookie 的值不为空，并且在指定的数据表中存在该记录，那么将直接进入网站的主页，不必进行登录操作；如果 Cookie 的值为空则输出网站登录页面，并且删除指定数据表中已经过期





的用户登录信息，其代码请参考技术要点的内容。

（2）创建 index_ok.php 文件，获取表单中提交的登录信息，完成登录操作。单点登录的第一个关键点：将登录成功的用户名和 Session ID 值保存到 Cookie 中，设置 Cookie 过期时间为 1 小时，有效范围是服务器的根目录。单点登录的第二个关键点：将登录的用户名和 Session ID 值存储到一个数据表中。这个 Cookie 值和数据表中的数据是验证当前用户是否具有单点登录权限的唯一依据，代码如下：

```
<?php
  session_start();
  include_once 'conn/conn.php';                          //执行连接数据库的操作
  $name = addslashes($_POST['name']);                    //获取用户名
  $pwd = $_POST['pwd'];                                  //获取密码
  if(!empty($name) and !empty($pwd)){
        $sql = "select * from tb_member where name = '".$name."' and password = '".$pwd."'";

        $num = $conne->getRowsNum($sql);                 //返回查询结果
        $conne->close_rst();                             //释放查询结果
        if($num == 0 or $num == ''){                     //如果不正确
              echo "<script>alert('用户名和密码不正确！ '); window.location.href='index.php';</script>";
        }else{                              //如果正确，则将登录用户名数据存储到 Cookie 中
              $session_id=session_id();                        //获取 SessionID
              setcookie("name", $name, time()+3600, "/");      //创建 Cookie
              setcookie("id", md5($session_id), time()+3600, "/");   //创建 Cookie
  $sqls="insert    into    tb_login(name,login_id,datetime)values('".$name."','".md5($session_id)."',
'".mktime()."')";        $nums = $conne->uidRst($sqls);        //返回查询结果
              echo "<script>alert('登录成功！ '); window.location.href='default.php';</script>";
        }
  }
?>
```

（3）创建 default.php 网站主页面。其关键就是根据 Cookie 值和指定数据表中存储的数据判断当前用户是否具有访问主页的权限。如果有，则输出网站主页的内容；否则，将跳转到登录页面。具体代码如下。

```
<?php
  if($_COOKIE[name]!="" and $_COOKIE[id]!=""){
        include_once 'conn/conn.php';                    //执行连接数据库的操作
        $sql = "select * from tb_login where name = '".$_COOKIE[name]."' and login_id = '".$_COOKIE
[id]."'";
        $num = $conne->getRowsNum($sql);                 //返回查询结果
        $conne->close_rst();                             //关闭数据库
        if($num == 0 or $num == ''){
              header("Location: index.php");
        }else{
?>
//省略了主页面代码
<?php
        }
```



```
    }else{
        header("Location: index.php");
    }
?>
```

（4）在 conn 文件夹下创建 conn.php 文件，实现连接数据库的操作。

技术要点

本实例开发了一个博客网站和一个论坛网站，通过 Cookie 实现单点登录的功能，即无论在博客或者论坛中只要登录后，这个用户名在一定时间内，在这两个网站中是通用的。

其关键就是对 Cookie 值的判断，如果 Cookie 的值存在并且与数据库中存储的值相同，那么就可以实现不同系统之间的跳转，否则就必须先登录，然后才可以访问系统中的内容。其关键代码如下：

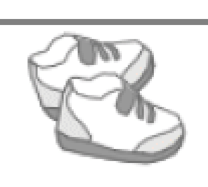
```
<?php
  include_once 'conn/conn.php';                          //执行连接数据库的操作
  if($_COOKIE[name]!="" and $_COOKIE[id]!=""){
        $sql = "select * from tb_login where name = '".$_COOKIE[name]."' and login_id = '".$_COOKIE
[id]."'";
        $num = $conne->getRowsNum($sql);                 //返回查询结果
        $conne->close_rst();                             //关闭数据库
        if($num != 0 or $num != ''){        //如果在数据表存在该记录，则直接跳转到登录页面
              header("Location: default.php");
        }else{                                           //否则输出登录页面的内容
?>
<!--省略了部分代码-->
<?php
        }
  }else{
  //清除数据库中 1 小时以前存储的用户登录数据（即 SESSION 过期以后的数据）
  $datetimes=mktime()-3600;                              //获取 1 小时前的时间戳
  $sqls="delete from tb_login where datetime<'".$datetimes."'";          //执行删除操作
  $nums = $conne->uidRst($sqls);                                         //返回查询结果
?>
   <!--省略了部分代码-->
<?php } ?>
```

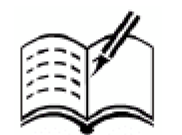
实例 065　统计用户在线时间

（实例位置：配套资源\SL\04\065）

实例说明

所谓用户在线时间就是当用户登录到页面开始，到关闭浏览器结束的时间差。而 Cookie 的生命周期在默认情况下关闭浏览器时自动删除，这非常符合在线统计时间的特点，是很多服务类网站优先选择的统计手段。





本实例通过创建 Cookie，保存用户登录到页面的时间戳，在用户单击“退出”按钮时，计算用户在线时间。运行结果如图 4.22 所示。

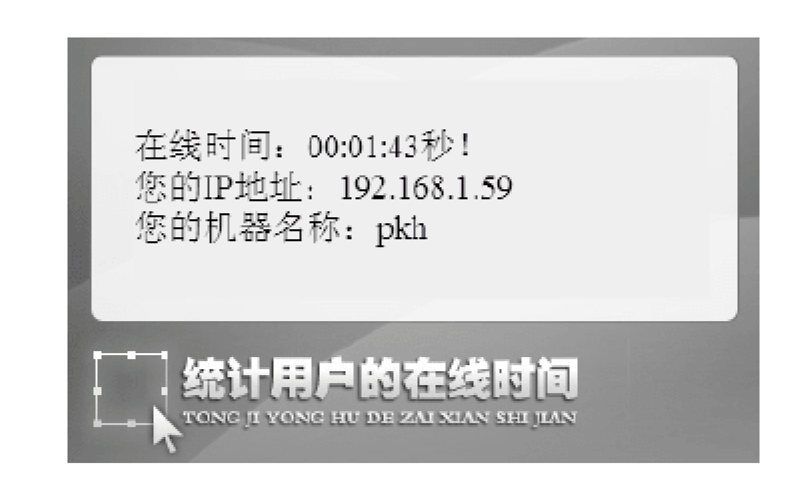


图 4.22　统计用户在线时间

实现过程

具体步骤如下：

（1）沿用实例 064 的 Forum 系统，在 index.php 文件中创建 Cookie，存储用户的登录时间戳。其关键代码如下：

```
<?php
  include_once 'conn/conn.php';                              //执行连接数据库的操作
  $str = time();
  setcookie('times',$str);
  if($_COOKIE[name]!="" and $_COOKIE[id]!=""){
        $sql = "select * from tb_login where name = '".$_COOKIE[name]."' and login_id = '".$_COOKIE
[id]."'";
        $num = $conne->getRowsNum($sql);               //返回查询结果
        $conne->close_rst();                                   //关闭数据库
        if($num != 0 or $num != ''){
              header("Location: default.php");
        }else{
?>
```

（2）创建 information.php 文件，用系统的当前时间戳减去 Cookie 中存储的登录时间戳，即获取到系统的在线时间的时间戳，并通过 date()函数对这个时间戳进行格式化，输出用户的在线时间，同时输出登录用户的 IP 和机器名称。其关键代码如下：

```
<?php
        $string = time()-$_COOKIE['times'];
        echo "在线时间：".date("H:i:s",$string)."秒！"."<br>";
        echo "您的 IP 地址：".getenv('REMOTE_ADDR')."<br>";
        echo "您的机器名称：".gethostbyaddr(getenv('REMOTE_ADDR'));
?>
```

技术要点

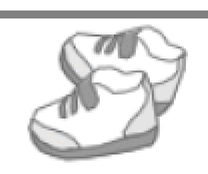
本实例的关键点是通过创建 Cookie 保存用户登录到页面的时间戳并与结束时间戳做减法运算。

用户登录时间：

```
$str = time();
  setcookie('times',$str);
```

用户退出时间：

```
$string = time()-$_COOKIE['times'];
  echo "在线时间：".date("H:i:s",$string)."秒！"."<br>";
```



实例 066　限制用户访问网站的时间

（实例位置：配套资源\SL\04\066）

实例说明

默认情况下 Cookie 生命周期是以关闭浏览器为基准。也就是说，如果用户不关闭浏览器，并且没有对 Cookie 进行设置，Cookie 永远也不会过期失效。假如用户在互联网发布网站，有成百上千的用户浏览这个网站，并且在线浏览用户数量一直增加。如果不对用户访问网站的时间进行限制，结果只能是服务器资源耗尽，网站瘫痪。

本实例通过设置 Cookie 限制用户访问网站的时间，运行结果如图 4.23 和图 4.24 所示。



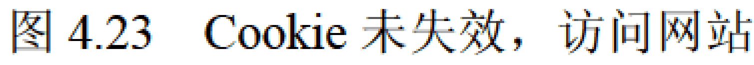


图 4.23　Cookie 未失效，访问网站



图 4.24　Cookie 失效

实现过程

具体步骤如下：

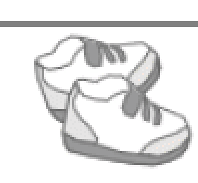
（1）创建 index.php 文件。首先，初始化 SESSION 变量，获取 Session_ID。然后，通过 setcookie()函数创建 Cookie，并将 Session_ID 作为 Cookie 值，同时设置 Cookie 的有效时间为 10 秒。

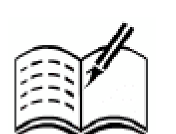
（2）在页面中通过判断 Cookie 变量的值是否为空来限制用户访问网站的时间。其关键代码如下：

```
<?php
  if(isset($_COOKIE['start'])|| $_COOKIE['start']==$session_id){
?>
…//省略了网页中的代码
<?php
  }else{
  echo "<h1 style ='color:red;'>您访问网站的时间到了</h1>";
}
?>
```

技术要点

本实例的关键点是通过 setcookie()函数创建 Cookie 并设置 Cookie 的有效时间。然后，通过判断 Cookie 变量是否存在和 Cookie 变量的值是否为空来控制访问网站的时间。其关





Note

键代码如下：

```
<?php
session_start();                              //初始化 SESSION 变量
$session_id=session_id();                     //获取 Session_ID
setcookie("start",$session_id,time()+10);     //定义 Cookie 变量，设置 Cookie 的有效时间是 10 秒
?>
```

实例 067　SESSION 更换聊天室界面

（实例位置：配套资源\SL\04\067）

实例说明

SESSION 可以实现数据在页面之间的传递，并且在 SESSION 的生命周期中一直有效。本实例中将运用 SESSION 的这个特性，编写一个简单的聊天室换肤功能。在聊天室中，根据提交的颜色值更换聊天室的背景颜色。其运行结果如图 4.25 所示。



图 4.25　更换聊天室背景颜色

实现过程

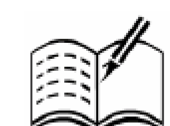
具体步骤如下：

（1）创建 index.php 文件，并设计一个简单的聊天室页面。在聊天室的 log 中插入一个颜色选项卡图片，并为每种颜色设置矩形热点链接，链接到 index.php 文件。同时，定义超链接参数 col，参数值是热点链接对应的颜色值。

（2）在 index.php 文件中，获取超链接传递的参数值，并将其赋给 SESSION 变量，然后根据 SESSION 变量的值设置页面中 body 标记的 bgcolor 的值，完成聊天室背景颜色的更换。index.php 的关键代码如下：

```
<?php
  $_SESSION['bgcolor']=$_GET['col'];    //获取超链接传递的参数值，并赋给 SESSION 变量
?>
<body bgcolor="<?php if($_SESSION['bgcolor']==""){echo "white";}else{echo $_SESSION
['bgcolor'];}?>">
//省略了部分代码
<map name="Map" id="Map">
<area shape="rect" coords="4,3,27,26" href="index.php?col=0066FF" />
<area shape="rect" coords="64,5,87,26" href="index.php?col=00CCFF" />
<area shape="rect" coords="31,34,55,56" href="index.php?col=999900" />
```





```
<area shape="rect" coords="218,64,237,85" href="index.php?col=CC9933" />
<area shape="rect" coords="184,33,209,55" href="index.php?col=CC6600" />
<area shape="rect" coords="214,7,237,28" href="index.php?col=3399FF" />
<area shape="rect" coords="8,63,27,87" href="index.php?col=996633" />
<area shape="rect" coords="65,61,88,87" href="index.php?col=99CC33" />
<area shape="rect" coords="152,61,177,87" href="index.php?col=CC3333" />
</map>
</body>
```

技术要点

本实例中，首先获取超链接中传递的参数值，然后将参数值赋给指定的 SESSION 变量，最后将 SESSION 变量设置为页面背景颜色 bgcolor 的值，进而实现聊天室背景颜色的更换。其关键代码如下：

```
<?php
 $_SESSION['bgcolor']=$_GET['col'];                //将超链接传递的参数值赋给 SESSION 变量
?>
 <body bgcolor="<?php if($_SESSION['bgcolor']==""){echo "white";}else{echo $_SESSION
['bgcolor'];}?>">
```

在 body 标记中，通过 if 语句判断 SESSION 变量的值。如果值为空，则设置 bgcolor 的值为“white”；否则，直接将 SESSION 变量作为 bgcolor 的值。

实例 068　掌控登录用户的权限

（实例位置：配套资源\SL\04\068）

实例说明

在论坛中，如果是管理员，则可以发布公告信息；如果是普通用户，就不可以发布公告信息。那么，程序中是如何对登录用户的权限进行判断的呢？这就是我们在本实例中将要向大家介绍的技术——通过 SESSION 变量掌控登录用户的权限。如果是管理员登录，则可以添加公告信息；如果是普通用户则只可以浏览论坛中的帖子。其运行结果如图 4.26 和图 4.27 所示。



图 4.26　管理员登录

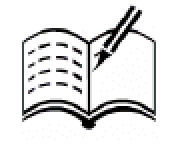


图 4.27　发布留言

如上所述，以管理员身份进行登录，进入到 main.php 主页后，可以单击“留言”超链



接，发布留言信息；如果是以普通用户身份登录，那么在 main.php 主页中，“留言”超链接是不可以点击的，因为普通用户不具备发布留言的权限。



Note

实现过程

具体步骤如下：

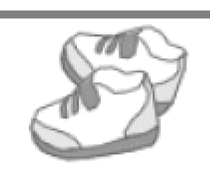
（1）创建 index.php 文件，添加表单，设计用户登录页面，并获取表单提交的用户名和密码，在本页中对用户提交的数据进行判断。首先，初始化 SESSION 变量，判断用户提交的值是否为空，如果登录用户是管理员，则为 SESSION 变量 type 赋值为 0；如果登录用户为普通用户，则为 SESSION 变量 type 赋值为 1。其关键代码如下：

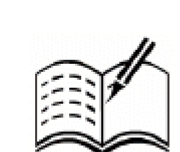
```
<?php
  session_start();
  if($_POST['sub']==true){
        if($_POST['user']!="" && $_POST['pwd']!=""){                   //判断用户名和密码是否为空
              if($_POST['user']=="mr" && $_POST['pwd']=="mrsoft"){       //判断是否为管理员
                    $_SESSION['type'] = 0;                              //为 SESSION 变量赋值为 0
                    echo "<script>alert('管理员登录成功！'); window.location.href='main.php';</script>";
              }else{
                    $_SESSION['type'] = 1;              //如果不是管理员，为 SESSION 变量赋值为 1
                    echo "<script>alert('用户名登录成功！'); window.location.href='main.php';</script>";
              }
              $_SESSION['user'] = $_POST['user'];    //将登录用户名存储到 SESSION 变量 user 中
              $_SESSION['pwd'] = $_POST['pwd'];     //将登录密码存储到 SESSION 变量 pwd 中
        }else{
              echo "<script>alert('用户名和密码不能为空！'); window.location.href='index.php';</script>";
        }
  }
?>
```

（2）创建 main.php 文件，并编写网站的主页面。首先初始化 SESSION 变量，然后根据 SESSION 变量 type 的值定义页面的输出内容。如果 SESSION 变量的值为 0，则说明登录用户是管理员，那么具备发布公告的权限；如果 SESSION 变量的值为 1，则说明登录用户是普通用户，就不具体发布公告的权限，即控制发布公告的超链接不输出。其关键代码如下：

```
<?php
session_start();                          //初始化 SESSION 变量
?>
<?php
  if($_SESSION['type']=="0"){//根据 SESSION 变量 type 的值决定是否输出发布公告的超链接
?>
<area shape="rect" coords="330,64,385,100" href="mes.php" />
<?php
  }
?>
```

（3）创建 mes.php 文件，编写发布公告信息的页面，根据 SESSION 变量 type 的值决定用户是否具有访问此页面的权限。





技术要点

（1）应用 SESSION，首先要初始化 SESSION 变量，其应用的是 session_start()函数，语法如下：

```
bool session_start ( void );
```

它判断是否有一个会话 ID 存在。如果不存在，就创建一个；如果存在，则将这个已注册的会话变量载入，以供使用。

（2）创建 SESSION 变量，应用预定义变量$_SESSION[]，语法如下：

```
$_SESSION["name"] = "value";
```

通过预定义变量$_SESSION[]定义 SESSION 变量。“name”为设置变量的名称，“value”表示设置变量的值。

实例 069　屏蔽页面刷新对计数器的影响

（实例位置：配套资源\SL\04\069　视频位置：配套资源\SP\04\069）

实例说明

计数器用来统计一个网站被访问的次数，它体现了一个网站的受关注程度。本实例将向大家介绍如何制作计数器，以及如何屏蔽刷新页面对计数器的影响。在本实例中，当用户访问时计数器的值会增加一次，但是无论如何刷新页面，计数器的值都不会再次增加，只有重新打开页面，计数器的值才会发生变化。本实例的运行结果如图 4.28 所示。

网站的访问量：　73

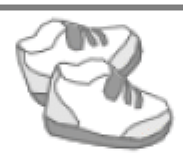
图 4.28　屏蔽页面刷新对计数器的影响

实现过程

具体步骤如下：

（1）创建 index.php 文件，并输出网页当前的访问量。首先，初始化一个 SESSION 变量。然后，判断 SESSION 变量的值是否为空，如果值为空，则打开指定的文本文件，读取其中存储的数据，并且将文本文件中的数据值增加 1，同时将新的数据写入到文本文件中，关闭文件。最后，为 SESSION 变量赋值为 1。其关键代码如下：

```
<?php session_start();
if($_SESSION[temp]==""){ //判断$_SESSION[temp]==""的值是否为空，其中的 temp 为自定义的变量
    if(($fp=fopen("counter.txt","r"))==false){
        echo "打开文件失败!";
    }else{
        $counter=fgets($fp,1024);          //读取文件中数据
        fclose($fp);                       //关闭文本文件
```



```
            $counter++;                        //计数器增加 1
            $fp=fopen("counter.txt","w");      //以写的方式打开文本文件<!---->
            fputs($fp,$counter);               //将新的统计数据增加 1
            fclose($fp);                       //关闭文件
        }
        $_SESSION[temp]=1; //登录以后，$_SESSION[temp]的值不为空，给$_SESSION[temp]赋一个值 1
    }
?>
```

（2）在 index.php 文件中，创建 img 标签，并在 src 属性中调用 gd1.php 文件，完成网站访问量的输出。

（3）创建 gd1.php 文件，通过文件系统函数读取存储在 counter.txt 中的网站访问量的数据，并通过 GD2 函数输出网站访问量的数据。

技术要点

SESSION 变量控制重复计数的原理如下：在当前页被访问时，初始化一个 SESSION 变量，判断 SESSION 变量的值是否为空。如果为空，则将计数器的值增加 1，并且为 SESSION 变量赋值为 1。此时，在当前页中 SESSION 变量的值已经不为空，无论如何刷新，SESSION 变量的值都不会改变，计数器的值也不会增加。

实例 070　SESSION 购物车

（实例位置：配套资源\SL\04\070　视频位置：配套资源\SP\04\070）

实例说明

购物车是在网上购物时使用的一个临时存储商品的“车辆”。购物车能为用户在网上购物提供很大的方便，不用担心一次购买多个商品时要进行多次提交结算的操作，可以将所购商品放入购物车中，等选购完所有商品之后，一起进行结算。在本实例中，将介绍 SESSION 购物车的实现方法，其运行结果如图 4.29 所示。



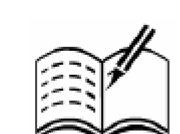
图 4.29　SESSION 购物车

实现过程

应用 SESSION 技术开发购物车，其具体步骤如下：

（1）创建 index.php 页面，实现商品展示功能。设计原理是：从数据库中读取商品信





息，将商品信息在页面中进行分栏、分页显示，并为商品设置购买和查看购物车的超链接，运行结果如图 4.30 所示。



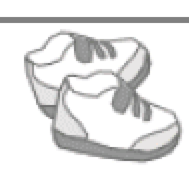
图 4.30　购物车商品展示页面

（2）创建 by_commodity.php 文件，实现购买功能，即向购物车中添加商品。首先创建一个购物车，然后判断购物车中是否为空。如果为空，则将商品展示页中“购买”超链接传递的商品 ID（$_GET[id]）和数量添加到购物车中；如果不为空，则判断添加商品的 ID 是否在购物车中已经存在，如果存在则不能重复添加，反之则将商品 ID 添加到购物车中，最后跳转到 shopping.php 购物车页面。其代码如下：

```
<?php
session_start();                                              //初始化 SESSION 变量
header ( "Content-type: text/html; charset=UTF-8" );          //设置文件编码格式
if($_SESSION["goodsid"]=="" && $_SESSION["goodsnum"]==""){//判断 SESSION 变量是否为空
    $_SESSION["goodsid"]=$_GET["id"]."@";//如果 SESSION 变量为空，则为其赋值为商品的
ID，并以@分隔
  $_SESSION["goodsnum"]="1@";        //如果 SESSION 变量为空，则为其赋值为 1，并以@分隔
}else{                                              //如果 SESSION 变量不为空
     $array=explode("@",$_SESSION["goodsid"]); //则以@为分隔符，将 SESSION 变量中的数据
写入到数组中
   if(in_array($_GET["id"],$array)){                  //如果判读数组中是否存在指定的 ID
     echo "<script>alert('该商品已经被放入购物车！');history.back();</script>";
     exit;
   }
    //如果数组中不存在指定的 ID，则说明该商品还没有放入购物车中
    $_SESSION["goodsid"].=$_GET["id"]."@";          //将该商品添加到购物车中
    $_SESSION["goodsnum"].="1@";                    //更改商品数量
}
echo "<script>window.location.href='shopping_car.php';</script>";
?>
```

（3）创建 shopping.php 购物车页面，对购买商品进行统计，包括购买数量、每种商品的金额和总金额，并且实现删除购物车中商品、更改购物车中商品数量以及清空购物车的功能，同时设置“继续购买”的超链接。其代码请参考本书配套资源。

（4）创建 change_commodity_counts.php 文件，实现更改购物车中商品数量的功能。首先获取 shopping.php 页面表单中提交的商品数量和商品 ID，然后通过正则表达式验证商品数量是否为正整数，最后根据商品 ID 找到指定的商品，更改商品数量，并返回到购物



车页面。其代码如下：

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
$id=$_POST["id"];                                            //获取商品 id
$num=$_POST["goodsnum"];                                     //获取商品数量
$preg="/^[0-9]*[0-9]$|^[0-9]*[0-9]$/";                       //编写正则表达式
if($num==""){                                                //判断提交的值是否为空
  echo "<script>alert('数量不能为空!');history.back();</script>";
  exit;
}else if(!preg_match($preg,$num,$str)){                      //判断提交的数据是否为正整数
  echo "<script>alert('数量只能为正整数!');history.back();</script>";
  exit;
}
$arrayid=explode("@",$_SESSION["goodsid"]);
$arraynum=explode("@",$_SESSION["goodsnum"]);
$key=array_search($id,$arrayid);          //在数组中搜索给定的值，如果成功则返回相应的键名
$arraynum[$key]=$num;                                        //更改商品数量
$_SESSION["goodsnum"]=implode("@",$arraynum);                //更改商品数量
echo "<script>window.location.href='shopping_car.php';</script>";
?>
```

（5）创建 delete_commodity.php 文件，删除购物车中指定的商品。首先获取超链接中传递的商品 ID，然后将 SESSION 中的字符串数据转换成数组，并根据商品 ID 获取指定要删除的商品的键名，最后根据键名为指定的数组赋空值，并重新将数组添加到购物车中。具体代码请参考本书配套资源。

（6）创建 clear_shopping_car.php 文件，清空购物车，即将 SESSION 变量$_SESSION ["goodsid"]和$_SESSION["goodsnum"]赋值为空。其代码如下：

```
<?php
session_start();                                       //初始化 SESSION 变量
header ( "Content-type: text/html; charset=UTF-8" );   //设置文件编码格式
$_SESSION["goodsid"]="";                               //为购物车的 ID 赋值为空
$_SESSION["goodsnum"]="";                              //为购物车数量赋值为空
echo "<script>window.location.href='shopping_car.php';</script>";//完成清空操作，跳转到购物页面
?>
```

技术要点

SESSION 购物车，顾名思义，其主要就是应用 SESSION 变量来实现的。而所谓的购物车，就是通过$_SESSION[]创建的两个 SESSION 变量：其中 goodsid 存储商品的 ID，goodsnum 存储商品的数量。

购物车的操作流程：首先，登录到网站中浏览商品。然后，购买指定的商品，进入到购物车页面中，可以实现很多操作，包括更改商品数量、删除商品、清空购物车、继续购物等。最后，填写收货人信息，生成订单，完成订单打印、预览，提交订单等操作。其操作流程如图 4.31 所示。



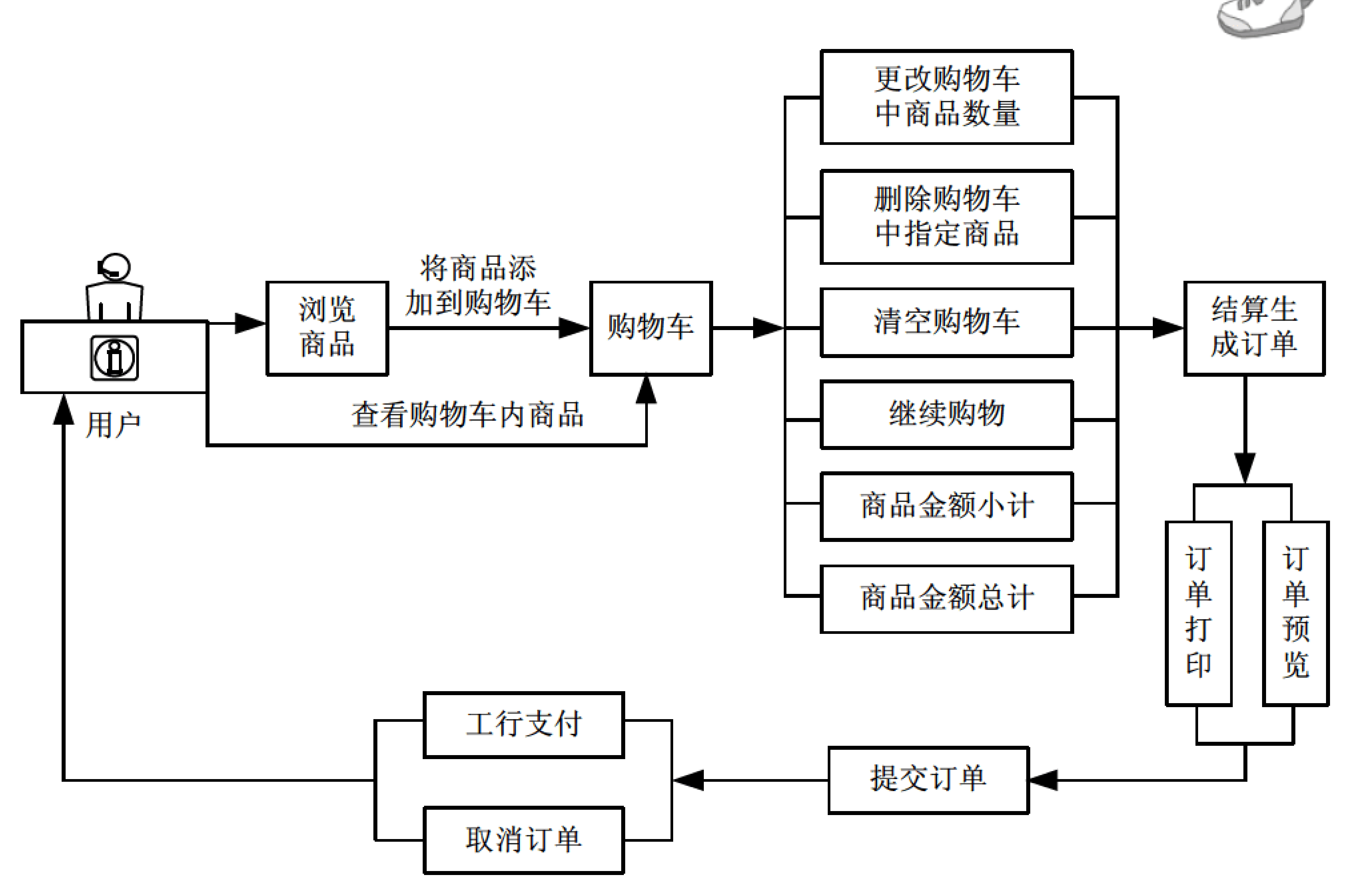


图 4.31 购物车操作流程

实例 071 清理 SESSION 缓存提高网站访问的效率

（实例位置：配套资源\SL\04\071）

实例说明

SESSION 缓存将网页中的内容临时存储到客户端 IE 的 Temporary Internet Files 文件夹下。当网页第一次被浏览后，页面的部分内容在规定的时间内就被临时存储在客户端的临时文件夹中。这样，下次访问这个页面时，就可以直接读取缓存中的内容，从而提高网站的浏览效率。但是，如果不对 SESSION 缓存做定期处理的话，也会给服务器带来压力。在本实例中，将讲解 SESSION 缓存的运用和清理的方法。其运行结果如图 4.32 所示。



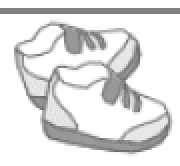
图 4.32 输出缓存信息

实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，定义 SESSION 临时缓存文件夹路径并开启缓存，设置缓存时间为 30 分钟，启用 SESSION，设置 SESSION 变量。其代码如下：

```
<?php
 $path =   './tmp/';                          //定义缓存文件的临时存储路径
 session_save_path($path);                   //设置缓存存储路径
 session_cache_limiter('private');           //设置缓存方式
```




```
    $session_cache = session_cache_limiter();        //开启缓存
    session_cache_expire(30);                        //定义缓存时间
    $session_expire = session_cache_expire();        //设置缓存时间
    session_start();                                 //初始化 SESSION 变量
    $_SESSION['cache'] = $session_cache;             //为 SESSION 变量赋值
    $_SESSION['expire'] = $session_expire;
?>
```

（2）在 index.php 文件中，获取并输出缓存中存储的数据，创建超链接，执行清理缓存的操作。清理方法是为$_SESSION 赋一个空数组值，并应用 session_destroy()函数彻底删除会话。其关键代码如下：

```
<?php
    echo "<a href = 'index.php?cache=0'>清理缓存</a>";
    if($_GET[cache] == "0"){
            $_SESSION = array();                     //清空 SESSION
            session_destroy();                       //销毁会话
            echo "<script>alert('清理 SESSION 缓存成功');location.href='http://www.mrbccd.com'</script>";
    }
?>
```

技术要点

在本实例中，运用 session_cache_limiter()函数创建缓存，应用 session_save_path()函数设置缓存文件的存储路径，应用 session_cache_expiry()函数设置缓存时间。

session_cache_limiter()函数用于创建 SESSION 缓存。其语法如下：

```
string session_cache_limiter ( [string cache_limiter])
```

参数 cache_limiter 用于设置缓存的方式。参数值的说明如表 4.3 所示。

表 4.3　cache_limiter 参数值的说明

参　数　值	说　　明
nochache	不设置缓存
private	私有方式
private nochache	私有方式，但不过期
public	公有方式

session_cache_expire ()函数用于设置缓存时间。其语法如下：

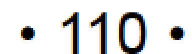
```
int session_cache_expire ( [int new_cache_expire])
```

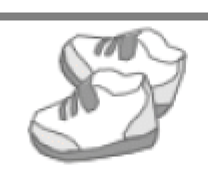
参数 cache_expire 为可选参数，用于设置 SESSION 的过期时间，单位为分钟。默认过期时间是 180 分钟（即不设置参数值的情况下）。

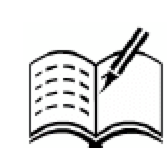
session_save_path()函数用于取得或者重新配置目前 SESSION 的存储路径。其语法如下：

```
string session_save_path([string path])
```

如果设置参数 path，表示重新设置 SESSION 的存储路径；如果不设置参数 path，表示直接获取当前 SESSION 的存储路径。







Note

实例 072　限制上传文件的大小

（实例位置：配套资源\SL\04\072　视频位置：配套资源\SP\04\072）

实例说明

在网站开发的过程中，为了确保能够充分地利用服务器的空间，在开发上传功能时最好能够对上传文件的大小进行控制。本实例开发的上传功能就可以对上传文件的大小进行限制。如果上传文件超过指定的范围，将给出提示信息，并终止上传。运行效果如图 4.33 所示。



图 4.33　限制上传文件的大小

实现过程

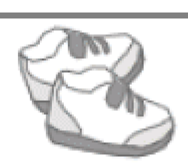
具体步骤如下：

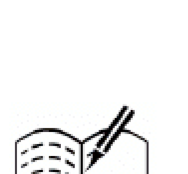
（1）创建 index.php 文件。

（2）添加表单，设置文件域、提交按钮，使用 POST 方法设置 enctype="multipart/form-data"，将数据提交到本页。

（3）通过$_FILES 获取上传文件的相关信息，通过 move_upload_file()函数完成图片上传。具体代码如下：

```
<?php
if(!empty($_FILES[up_picture][name])){                          //判断上传内容是否为空
  if($_FILES['up_picture']['error']>0){                         //判断文件是否可以上传到服务器
        echo "上传错误:";
        switch($_FILES['up_picture']['error']){
              case 1:
                    echo "上传文件大小超出配置文件规定值";
              break;
              case 2:
                    echo "上传文件大小超出表单中约定值";
              break;
              case 3:
                    echo "上传文件不全";
              break;
              case 4:
                    echo "没有上传文件";
              break;
        }
  }else{
        if(!is_dir("./upfile/")){                               //判断指定目录是否存在
              mkdir("./upfile/");                               //创建目录
        }
        $path='./upfile/'.time().strstr($_FILES['up_picture']['name'],'.'); //定义文件名称和存储位置
```





```
            if(is_uploaded_file($_FILES['up_picture']['tmp_name'])){      //是否是 HTTP POST 上传
                if(!move_uploaded_file($_FILES['up_picture']['tmp_name'],$path)){ //执行上传
                    echo "上传失败";
                }else{
                    echo "文件".time().$_FILES['up_picture']['name']."上传成功，大小为：".$_FILES
['up_picture']['size'];
                }
            }else{
                echo "上传文件".$_FILES['up_pictute']['name']."不合法！";
            }
        }
    }
    ?>
```

技术要点

控制上传文件的大小有两个关键点：第一点是 PHP 的配置文件 php.ini 中对上传文件的限制，如果上传文件超过它指定的范围，那么上传就会失败；第二点是在 PHP 配置文件允许的范围内，在程序中对上传文件大小的限制。

（1）PHP 中通过 php.ini 文件对上传文件进行限制，包括：是否支持上传、上传文件的临时目录、上传文件的大小、指令执行的时间、指令分配的内存空间。

在 php.ini 中，定义到 File Uploads 项，完成对上传相关选项的设置。上传相关选项的含义如下：

- ☑ file_uploads：如果值是 on，说明服务器支持文件上传；如果为 off，则不支持。一般默认是支持的，这个不用修改。
- ☑ upload_tmp_dir：上传文件临时目录。在文件被成功上传之前，文件首先存放到服务器端的临时目录中。多数使用系统默认目录，但是也可以自行设置。
- ☑ upload_max_filesize：服务器允许上传文件的最大值，以 MB 为单位。系统默认为 2MB，如果网站需要上传超过 2MB 的数据，那么就要修改这个值。

上述是 php.ini 中 File_Uploads 项中与上传相关选项参数设置的说明。除了 File_Uploads 项中的内容外，在 php.ini 中还有其他几个选项会影响到文件的上传。

- ☑ max_execution_time：PHP 中一个指令所能执行的最大时间，单位是秒。该选项在上传超大文件时必须修改，不然，即使上传文件在服务器允许的范围内，但是超过了指令所能执行的最大时间，也无法实现上传。
- ☑ memory_limit：PHP 中一个指令所分配的内存空间，单位是 MB。它的大小同样会影响到超大文件的上传。

（2）在客户端限制上传文件，应用的是 form 表单中的 enctype 和 method 属性，以及隐藏域 MAX_FILE_SIZE。

- ☑ enctype="multipart/form-data"：指定表单编码数据方式。
- ☑ method="post"：指定数据的传输方式。
- ☑ <input type="hidden" name="MAX_FILE_SIZE" value="10000" />：通过隐藏域限制上传文件的大小（单位为字节），该值不能超过 php.ini 配置文件中 upload_max_filesize 选项设置的值。它不能完全控制上传文件的大小，只是可以避免一些不必





Note

要的麻烦。切记，它只有放置在 file 文件域之前，才能有效果。

上述两种判断的结果都可以通过全局变量$_FILES 的返回值来体现。$_FILES 是一个数组，它包含所有上传的文件信息。下面介绍一下$_FILES 数组中每个元素的含义，如表 4.4 所示。



表 4.4　$_FILES 数组中元素

元　素　名	说　　明
$_FILES[filename][name]	存储上传文件的文件名，如 text.txt、title.jpg 等
$_FILES[filename][size]	存储文件的大小，单位为字节
$_FILES[filename][tmp_name]	存储文件在临时目录中使用的文件名。文件在上传时，先要将其以临时文件的身份保存在临时目录中
$_FILES[filename][type]	存储上传文件的 MIME 类型，MIME 类型规定各种文件格式的类型。每种 MIME 类型都是由“/”分隔的主类型和子类型组成。例如，“image/gif”，主类型为“图像”，子类型为 GIF 格式的文件，“text/html”代表文本的 HTML 文件
$_FILES[filename][error]	存储与文件上传相关的错误代码。此项目是 PHP 4.2.0 版本中新增的内容，其返回值有 5 种，如下所示： 0：表示没有任何错误，文件上传成功 1：表示上传文件的大小超出了 PHP 配置文件指令 upload_max_filesize 选项限制的值 2：表示上传文件大小超出了 HTML 表单中 MAX_FILE_SIZE 选项所指定的值 3：表示文件只被上传了一部分 4：表示没有上载任何文件

实例 073　限制上传文件的类型

（实例位置：配套资源\SL\04\073）

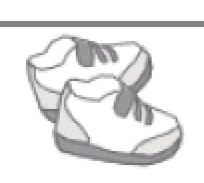
实例说明

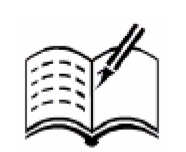
在开发文件上传功能时，不仅要考虑上传文件的大小，对上传文件的类型也要考虑。针对不同的需要，对上传文件的类型也要限制，这里指的文件类型可以通过文件的扩展名（例如，.txt、.php、.doc）来判断。如在进行文件上传时，在涉及到文本说明性的文字时，最好将上传的文件限制以“.txt”为后缀的文本格式。

运行本实例，单击图中的“浏览”按钮，选择要上传的文件（文件不是以“.txt”为扩展名），单击“提交”按钮程序将给出错误提示，如图 4.34 所示。



图 4.34　限制上传文件的类型





Note

实现过程

继续使用实例072的内容，在此基础上增加对上传文件类型的判断，即通过预定义变量$_FILES获取上传文件的名称，同时截取上传文件名称的后缀并加以判断。如果后缀部分与“.txt”相匹配，则执行上传操作；否则，提示用户上传类型不正确。增加的关键代码如下：

```
<?php
if(!empty($_FILES['up_picture'][ 'name'])){     //判断上传内容是否为空
$type=$_FILES['up_picture']['name'];           //获取上传文件的名称
$types=strstr($type,'.');                      //截取上传文件的后缀
if($types==".txt"){                            //判断上传文件的后缀是否符合要求
        //省略了部分代码
  }
  }else{
                echo "上传文件".$_FILES['up_pictute']['name']."类型不正确！ ";
  }
}
?>
```

技术要点

文件的上传和对大小的限制已经在前面的实例中讲解过，这里只讲解限制上传文件类型的关键技术。首先通过预定义全局变量$_FILES中的$_FILES['files']['name']获取上传文件的名称，然后应用strstr()截取上传文件名称的后缀，最后判断上传文件的后缀是否符合要求。

strstr()函数的语法如下：

```
string strstr(string strings, string needle);
```

该函数用于在字符串strings中查找字符串needle的位置，并返回从needle字符串开始到strings字符串结束所有的内容。如果在strings中没有查找到needle，该函数将返回false。

多学两招：

判断上传文件是否符合多种类型中的一种，最佳的方法就是将多种类型定义到一个数组中，然后通过for循环输出数组中的元素值，将输出的元素值与获取的文件后缀进行比较，进而判断其是否符合要求。

实例074　通过链接方式下载

（实例位置：配套资源\SL\04\074）

实例说明

在一个完整的网站中，拥有文件下载的功能能够给网站带来更多的访问者，增加网站



的访问量。最常用的文件下载方式就是通过链接下载。在本实例中，应用链接方式进行下载，当单击“下载”文字的超链接或单击右键“另存为”时，即可弹出文件下载的对话框，完成下载操作。其运行结果如图 4.35 所示。

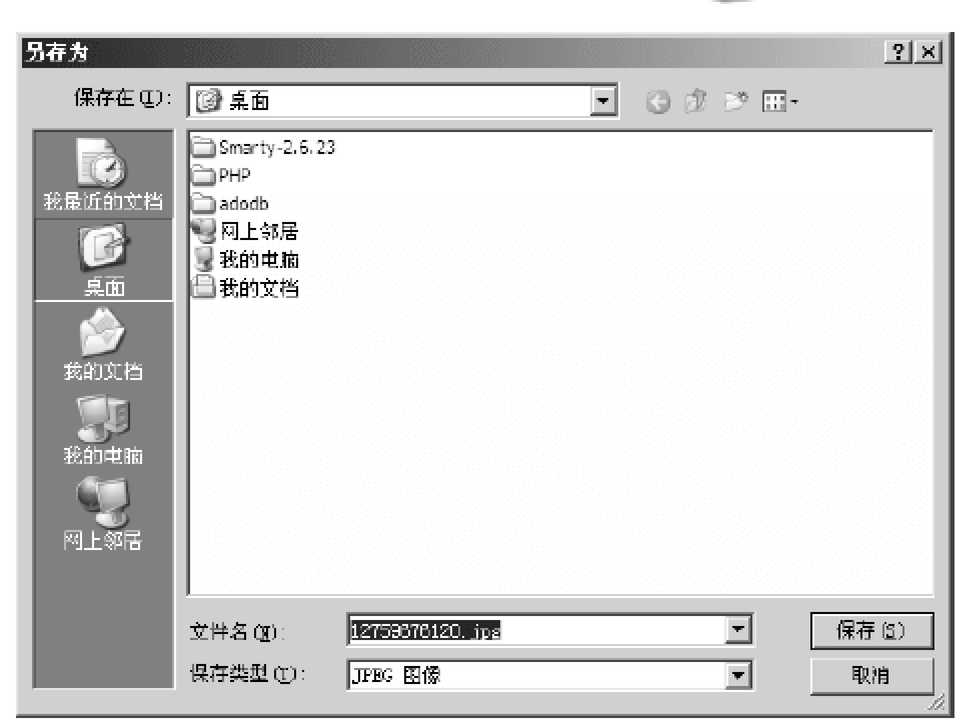


图 4.35　通过链接方式下载

实现过程

首先创建一个多文件上传页面 index.php，然后为输出的图片名称创建超链接。当用户在该超链接名称上单击鼠标右键并选择“另存为”时，即可弹出文件下载的对话框，选择完路径后单击“保存”按钮，完成图片的下载操作。其关键代码如下：

```
<a href="<?php echo $myrow[file_test];?>">
<?php echo $myrow[file_name];?></a>
```

技术要点

本实例主要通过一个超链接来实现文件的下载。通过 select 语句从数据库中读取文件存储的路径，然后将该文件作一个超链接，当单击“下载”超链接时，即可下载该文件。

多学两招：

如果是压缩的文件可以直接下载；如果不是，则需要单击鼠标右键，在弹出的快捷菜单中选择“另存为”命令。

实例 075　上传多个文件到服务器

（实例位置：配套资源\SL\04\075）

实例说明

上传图片到服务器是程序开发过程中必不可少的一个功能。它不但可以达到图片共享的目的，而且可以提高网站的访问量，丰富网站的内容。本实例讲解如何通过 POST 方式实现多图片上传，运行结果如图 4.36 所示。

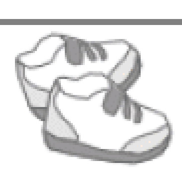


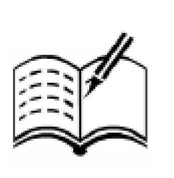
图 4.36　POST 方式多图片上传

实现过程

具体步骤如下：

（1）创建 index.php 文件。添加表单，设置文件域、提交按钮，使用 POST 方法，设置 enctype="multipart/form-data"，将数据提交到 index_ok.php 页，完成多个文件的上传操





Note

作。其关键代码如下：

```
<table width="750" border="0" cellspacing="0" cellpadding="0">
    <form action="index_ok.html" method="post" enctype="multipart/form-data" name="form1">
        <tr>
          <td width="100" height="25" align="right" class="STYLE1">内容 1：</td>
          <td  width="150"  align="center"><input  name="files[]"  type="text"  id="files[]"  size=
"15"></td>
          <td align="left"><input name="picture[]" type="file" id="picture[]" size="30"></td>
          </tr>
        <tr>
          <td height="25" align="right" class="STYLE1">内容 2：</td>
          <td align="center"><input name="files[]" type="text" id="files[]" size="15"></td>
          <td align="left"><input name="picture[]" type="file" id="picture[]" size="30"></td>
          </tr>
        <tr>
          <td colspan="3" align="center">
              <input  type="image"  name="imageField"  src="images/bg_09.jpg">  
  
              <input type="image" name="imageField2" src="images/bg_11.jpg"></td>
          </tr>
      </form>
    </table>
```

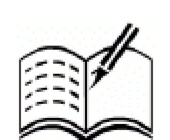
（2）在 index.php 文件中，连接数据库，读取数据库中存储的数据，实现上传文件的分页输出。具体代码请参考配套资源中的内容。

（3）创建 index_ok.php 文件，获取表单中提交的数据，将多个文件存储到服务器中，并将文件的名称和存储路径存储到数据库中。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );            //设置文件的编码格式
include "conn/conn.php";                                        //包含数据库链接文件
if ($_POST ["files"] != "") {
  if (! is_dir ( "./upfile" )) {
        mkdir ( "./upfile" );                                   //创建上传文件的存储文件夹
  }
  $data = date ( "Y-m-d H:m:s" );                               //定义时间
  function check($var) {                                        //验证数组的返回值是否为空
  return ($var != "");                                          //如果不为空，则返回数组元素
}
$files = array_filter ( $_POST ["files"], "check" );            //去除数组中的空值
$array = array_filter ( $_FILES ["picture"] ["name"], "check" );          //去除数组中的空值
foreach ( $array as $key => $value ) {              //循环读取数组中的数据
  $path = 'upfile/' . time () . $key . strtolower ( strstr ( $value, "." ) );      //定义上传文件的存储位置
  move_uploaded_file ( $_FILES ["picture"] ["tmp_name"] [$key], $path );  //执行上传操作
  $query = "insert into tb_up_file (file_test,data,file_name) values ('$path','$data','$files[$key]')";
  $result = mysql_query ( $query );
}
```



```
    echo "<script>alert('图片上传成功'); window.location.href='index.html';</script>";
    }
    ?>
```



技术要点

多文件上传的关键是如何定义上传文件元素的名称，以及如何判断上传文件的数量。在本实例中，以数组的形式定义上传文件的名称（上传文件的名称是“files[]”）。为了达到可以上传任意数量图片（4 个图片以内）的目的，在对上传文件进行处理的过程中，应用 array_filter()函数和回调函数去除数组中的空元素。

array_filter()函数，表示用回调函数过滤数组中的单元，语法如下：

```
    array array_filter ( array input [, callback callback] )
```

array_filter()函数依次将 input 数组中的每个值传递到 callback 函数。如果 callback 函数返回 true，则 input 数组的当前值会被包含在返回的结果数组中，并且数组的键名保持不变。

本实例中定义的回调函数是 check()，用于验证数组中的返回值是否为空。其语法如下：

```
    function check($var) { //验证数组的返回值是否为空
            return ($var != "");
      }
```

指点迷津：

在回调函数中，不要对数组进行修改操作，例如，增加或者删除数组中的元素。如果数组一旦改变，那么此函数的运用也就没有意义了。如果没有提供 callback()函数，那么 array_filter()将删除 input 中所有值为 false 的元素。

通过 POST 方法实现多图片上传，在创建 form 表单时，必须指定 enctype="multipart/form-data"属性。如果要通过隐藏域 MAX_FILE_SIZE 的值对上传文件的大小进行限制，那么必须将隐藏域放置在上传文件的文件域之前，否则是不会起到作用的。

多学两招：

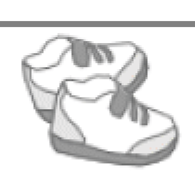
通过伪静态技术隐藏 PHP 文件后缀，是通过修改 Apache 服务器的配置文件 httpd.conf 来完成的。首先，将#LoadModule rewrite_module modules/mod_rewrite.so 前面的“#”去掉，并启动该项。然后，查找 httpd.conf 文件，找到其中的“AllowOverride”项，将它的值都改为 All，保存并重新启动 Apache 服务器，即可使修改生效。最后，在实例根目录下创建.htaccess 文件，实现对 PHP 文件后缀的隐藏操作。

实例 076　通过 header()函数进行下载

（实例位置：配套资源\SL\04\076　视频位置：配套资源\SP\04\076）

实例说明

除了可以通过链接方式下载文件之外，还可以通过 header()函数完成下载的操作。例



如，本实例就是应用 header()函数实现文件的下载，如果下载的文件不存在，则给出提示信息。运行结果如图 4.37 所示。

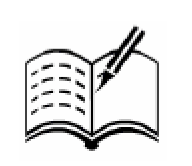






图 4.37　header()函数文件下载

实现过程

具体步骤如下：

（1）在实例 074 的基础上，添加单击图片下载的功能。其实现过程是为输出的图片创建超链接，链接到 download.php 文件，再利用 header()函数通过 download.php 完成文件下载的操作。在 index.php 文件中所做的修改如下：

```
<td width="245" height="100" align="center" valign="middle" bgcolor="#F0F0F0"><a href="download-<?php echo $myrow[file_test];?>.html" title="点击即可下载！"><?php echo "<img src=\"$myrow[file_test]\" width=\"250\" height=\"100\" border=\"0\">"?></a></td>
```

（2）创建 download.php 文件，应用 header()函数实现文件的下载。其代码如下：

```
<?php
 header("Content-type: text/html; charset=UTF-8");                //设置文件编码格式
 $path = $_GET['path'];                                            //获取文件路径
 if(!empty($path) and !is_null($path)){                            //判断变量是否为空，是否为 NULL
        $filename=basename($path);                                 //获取文件名
        $file=@fopen($path,"r");
        if($file){
               header("Content-type:application/octet-stream");    //输出 MIME 类型
               header("Accept-ranges:bytes");                      //接受的范围单位
               header("Accept-length:".filesize($path));           //文件长度
               header("Content-Disposition:attachment;filename=".$filename);  //缺省时文件保存对话框中的文件名称
               echo fread($file,filesize($path));                  //读取文件
               fclose($file);                                      //关闭文件
               exit;                                               //退出
        }else{
               echo "<script>alert('您下载的文件不存在！'); history.back();</script>";
        }
 }
?>
```

技术要点

通过 HTTP 方式下载文件，主要应用 header()函数。





header()函数属于 HTTP 函数。其作用是以 HTTP 协议将 HTML 文档的标头送到浏览器，并告诉浏览器具体怎么处理这个页面。header()函数的语法如下：

```
void header ( string string [, bool replace [, int http_response_code]] )
```

参数说明：

☑ string：发送的标头。

☑ replace：如果一次发送多个标头，对于相似的标头是替换还是添加。如果是 false，则强制发送多个同类型的标头。默认是 true，即替换。

☑ http_response_code：强制 HTTP 响应为指定值。

通过 HTTP 下载的代码如下：

```
header("Content-type: application/x-gzip");
header("Content-Disposition: attachment; filename=文件名");
header("Content-Description: PHP3 Generated Data"); >
```

HTTP 标头有很多，这里介绍的是下载的 HTTP 标头。其代码如下：

```
header('Content-Disposition: attachment; filename="filename"');
```

在应用的过程中，唯一需要改动的就是 filename，即将 filename 替换为要下载的文件。

多学两招：

通过 header()函数，不但可以实现文件的下载，还可以实现下面 3 个功能：

（1）重定向，这是最常用的功能。

```
header("Location: http://www.mrbccd.com");
```

（2）强制客户端每次访问页面时获取最新资料，而不是使用存储于客户端的缓存。

```
//设置页面的过期时间(用格林威治时间表示)。
header("Expires:     Mon,     08     Jul     2008     08:08:08     GMT");
//设置页面的最后更新日期(用格林威治时间表示)，使浏览器获取最新资料
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . "GMT");
header("Cache-Control: no-cache,     must-revalidate");          //控制页面不使用缓存
header("Pragma: no-cache");                    //参数（与以前的服务器兼容）,即兼容
HTTP1.0 协议
header("Content-type:     application/file");                    //输出 MIME 类型
header("Content-Length:     227685");                            //文件长度
header("Accept-Ranges:     bytes");                              //接受的范围单位
//默认为文件保存对话框中的文件名称
header("Content－Disposition: attachment; filename=$filename");    //实现下载
```

（3）输出状态值到浏览器，控制访问权限。

```
header('HTTP/1.1 401 Unauthorized');
header('status: 401 Unauthorized');
```





实例 077　重新定义上传文件的名称

（实例位置：配套资源\SL\04\077）

实例说明

应用 POST 方法上传文件时，需要将上传文件保存到服务器指定的目录中，这时可能会出现因名称相同而文件相互替换的情况。为了解决上述问题，可以应用 basename()函数和随机函数 mt_rand()对上传文件进行重新命名。在运行本实例时，如果用户成功上传重定义名称后的文件，将弹出新的文件名称对话框，如图 4.38 所示。



图 4.38　重新定义上传文件的名称

实现过程

在本实例中，关键是在定义上传文件的名称时应用随机函数 mt_rand()、时间戳以及 basename()函数。其关键代码如下：

```
$date=date("YmdHis");                                          //定义随机数
$filename=mt_rand(1000,9999).$date.basename($_FILES['up_picture']['name']);//定义上传文件名称
echo "<script>alert('文件名被替换为".$filename."');</script>";
$path='./upfile/'.$filename;                                   //定义上传文件名称和存储位置
```

技术要点

应用本实例也可以实现多文件上传的功能，但并非本实例中的技术重点，故不再赘述。这里将介绍一种给上传文件命名的方法。

basename()函数用于返回指定文件目录中的基本文件名，语法如下：

```
string basename(string path [, string suffix])
```

参数说明：

☑ path：指定文件的路径。

☑ suffix：可选参数，如果文件路径以 suffix 结尾，那么这部分内容被去掉。

通过这个函数获取到上传文件的原始名称，并对这个名称进行重新定义，进而避免在将文件上传到服务器后出现重名的问题。

多学两招：

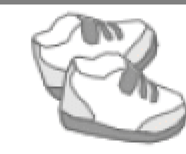
在通过表单中的文件域提交上传文件时，如果在 form 中定义了“enctype="multipart/form-data"”属性，那么在获取上传文件的原始名称时必须使用$_FILES 全局变量，此时如果使用$_POST 是获取不到值的。



MySQL 数据库

本章读者可以学到如下实例：

- 实例 078　启动 MySQL 服务
- 实例 079　连接 MySQL 服务
- 实例 080　关闭 MySQL 服务
- 实例 081　创建 PHP 图书数据库
- 实例 082　选择 PHP 图书数据库
- 实例 083　删除 PHP 图书数据库
- 实例 084　在 PHP 图书数据库中创建图书信息表
- 实例 085　查看图书信息表
- 实例 086　修改图书信息表
- 实例 087　重命名图书信息表
- 实例 088　删除图书信息表
- 实例 089　向图书信息表中添加数据
- 实例 090　修改图书信息表中的数据
- 实例 091　删除图书信息表中所有数据
- 实例 092　删除图书信息表中指定数据
- 实例 093　通过 phpMyAdmin 修改 MySQL 用户密码
- 实例 094　通过 phpMyAdmin 设置数据库、数据表编码
- 实例 095　phpMyAdmin 操作数据库
- 实例 096　phpMyAdmin 操作数据表
- 实例 097　phpMyAdmin 操作数据



实例 078　启动 MySQL 服务

（实例位置：配套资源\SL\05\078　视频位置：配套资源\SP\05\078）

实例说明

如果想学习 PHP 语言，那么就必须学习 MySQL。PHP 与 MySQL 是最佳的组合，虽然 PHP 如今可支持很多的数据库，如 Access、SQL Server、Oracle、DB2 等，但是无论在 LAMP 组合（LINUX + Apache + MySQL + PHP）还是 AMP 组合（Apache + MySQL + PHP）中，MySQL 的地位丝毫没有动摇。

在 Windows 操作系统中，启动 MySQL 服务可以使用命令模式进行。

实现过程

在 Windows 操作系统下启动 MySQL 服务的步骤如下：

（1）选择“开始”/“运行”命令，在弹出的运行窗口中输入“cmd”命令，按 Enter 键进入命令提示符窗口，如图 5.1 所示。

（2）在命令提示符下输入启动 MySQL 服务的命令，如图 5.2 所示。



图 5.1　启动 MySQL 服务的第一步



图 5.2　成功启动 MySQL 服务

技术要点

本实例的关键是在命令提示符下输入启动 MySQL 服务的命令，完成 MySQL 服务的启动。其命令如下：

```
net start mysql
```

实例 079　连接 MySQL 服务器

（实例位置：配套资源\SL\05\079　视频位置：配套资源\SP\05\079）

实例说明

启动 MySQL 服务只是应用 MySQL 的第一步，如果想要操作 MySQL 数据库，就要学会连接 MySQL 服务器。本实例将详细讲解在命令提示符下如何连接 MySQL 服务器。

实现过程

连接 MySQL 服务器的步骤如下：

（1）进入命令提示符窗口。





（2）如果 MySQL 默认安装在 C:\盘根目录下，在未设置环境变量的情况下，需要在命令提示符下进入“mysql”\“bin”文件夹，输入命令。假设用户名为 root，密码为 111，输入的命令如下：

```
mysql –uroot –p111
```

按 Enter 键，输出如图 5.3 所示的内容，表示连接 MySQL 服务器成功。

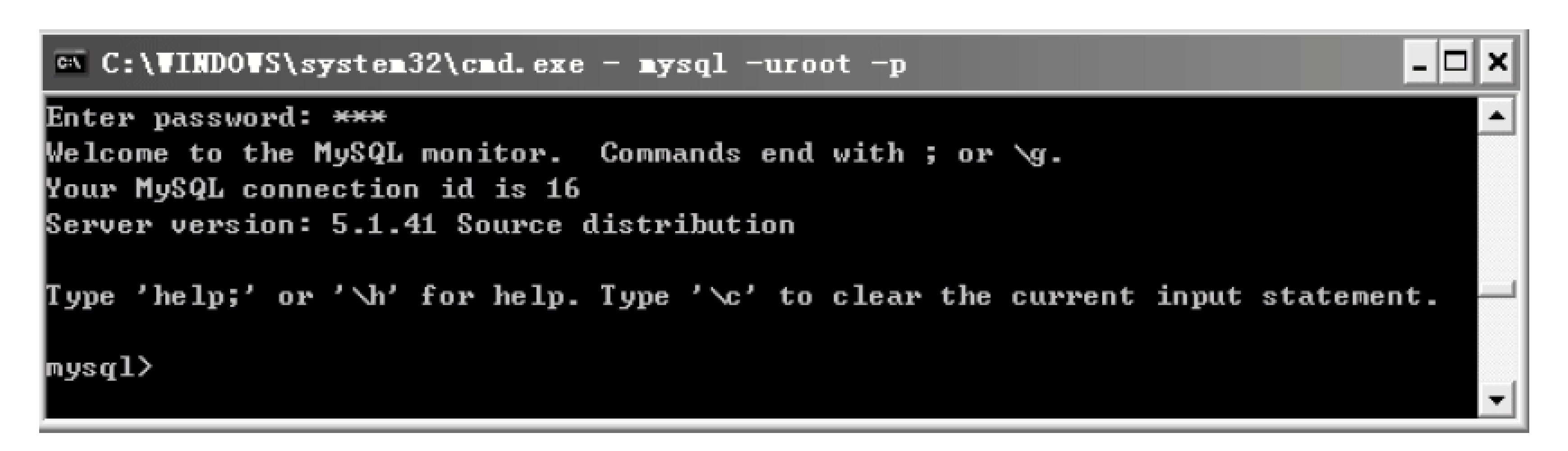


图 5.3　连接 MySQL 服务器成功

（3）如果在本地用户机上运行程序，可以直接输入命令“mysql –uroot –p111”来连接 MySQL 服务器，但是为了安全起见，避免他人得到连接 MySQL 服务器的密码进行非法操作，最好使用如下命令：

```
mysql –uroot –p
```

根据提示输入密码，完成连接，如图 5.4 所示。



图 5.4　连接 MySQL 服务器成功

技术要点

选择“开始”/“运行”命令，在弹出的窗口中输入“cmd”命令，进入命令提示符提示框。输入如下命令：

```
mysql –uName –p Password
```

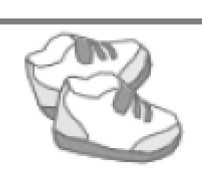
“mysql”与“-u”和“Name”和“-p”之间有一个空格。Name 表示进入 MySQL 服务器的用户名。Password 表示进入 MySQL 服务器的密码。

实例 080　关闭 MySQL 服务

（实例位置：配套资源\SL\05\080　视频位置：配套资源\SP\05\080）

实例说明

为了有效节省系统资源，在进行 MySQL 数据库操作后，需要及时将 MySQL 服务关闭。关闭 MySQL 服务与启动 MySQL 服务相类似，可以通过两种方法实现关闭 MySQL 服务。通过本实例，相信用户会将关闭 MySQL 服务的两种方法熟记于心中。





Note



实现过程

具体步骤如下：

（1）在命令提示符下，输入命令连接 MySQL，通过 MySQL 数据库函数“user”显示数据表中的内容，如图 5.5 所示。



图 5.5　查询表信息

（2）通过命令“exit”断开与 MySQL 的连接，如图 5.6 所示。

（3）输入命令“net stop mysql”，按 Enter 键，关闭 MySQL 服务，如图 5.7 所示。

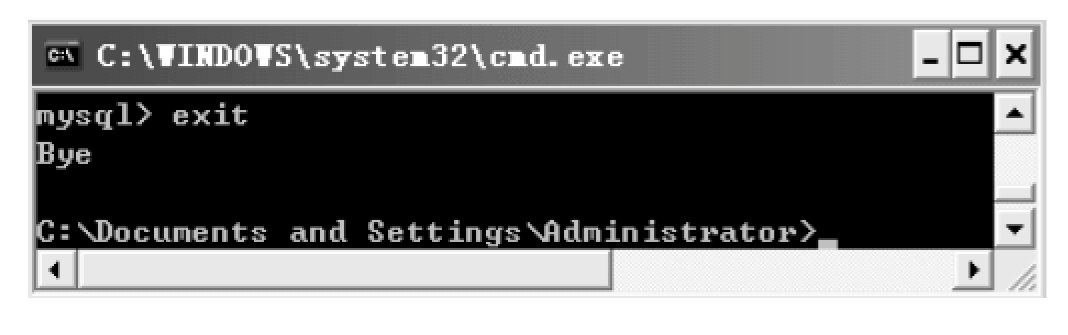


图 5.6　断开与 MySQL 连接

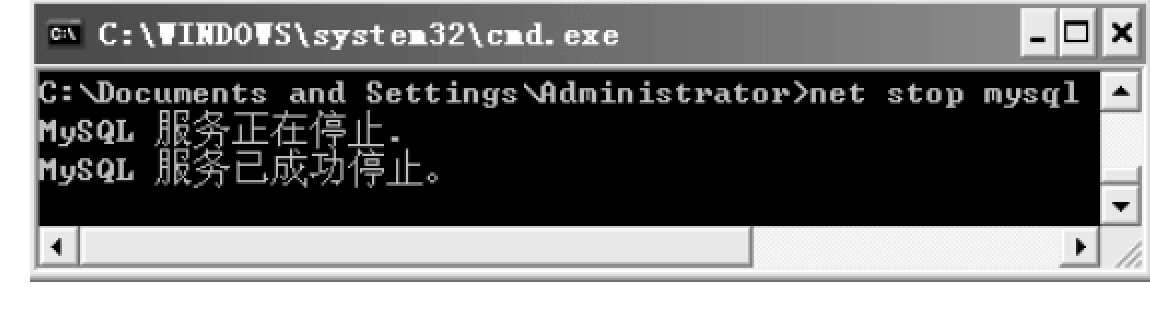


图 5.7　关闭 MySQL 服务

技术要点

命令提示符下，输入命令：

（1）断开 MySQL 数据库连接。

```
exit
```

（2）关闭 MySQL 服务。

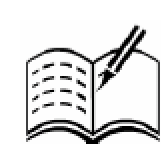
```
net stop mysql
```

实例 081　创建 PHP 图书数据库

实例说明

如果用户已经掌握如何启动 MySQL 服务、连接 MySQL、断开 MySQL 和关闭 MySQL 服务。下面就走进 MySQL 数据库，学习如何创建数据库。创建数据库的主要目的是将同类的或者是同一个项目中的数据表，进行分类存储，方便开发人员查找和使用。本实例通过数据库函数“CREATE DATABASE”创建 PHP 图书数据库。





实现过程

具体步骤如下：

（1）进入命令提示符窗口，启动 MySQL 服务，连接 MySQL 服务器。

（2）通过数据库函数在命令提示符下编写命令。代码如下：

```
CREATE DATABASE DB_MRBOOK;
```

按 Enter 键，运行结果如图 5.8 所示。

（3）查看 MySQL 数据库列表，查找创建的 PHP 图书数据库是否存在，运行结果如图 5.9 所示。

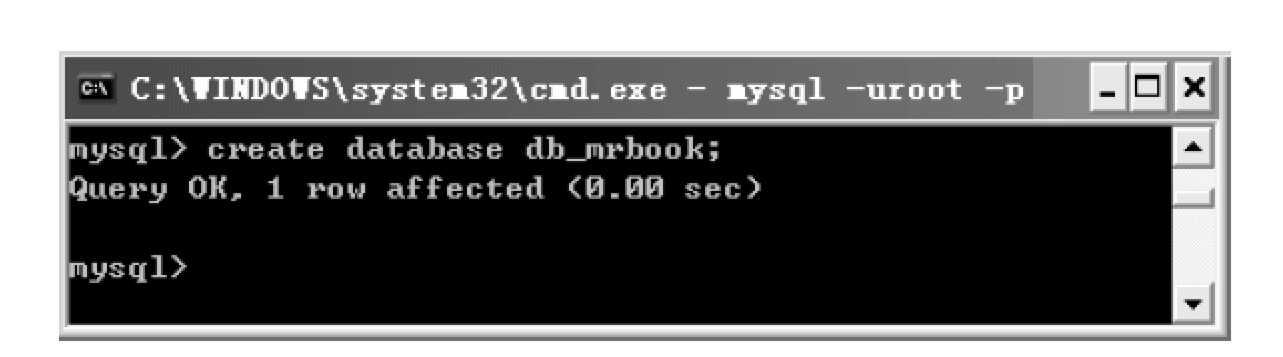




图 5.8　创建数据库 db_mrbook

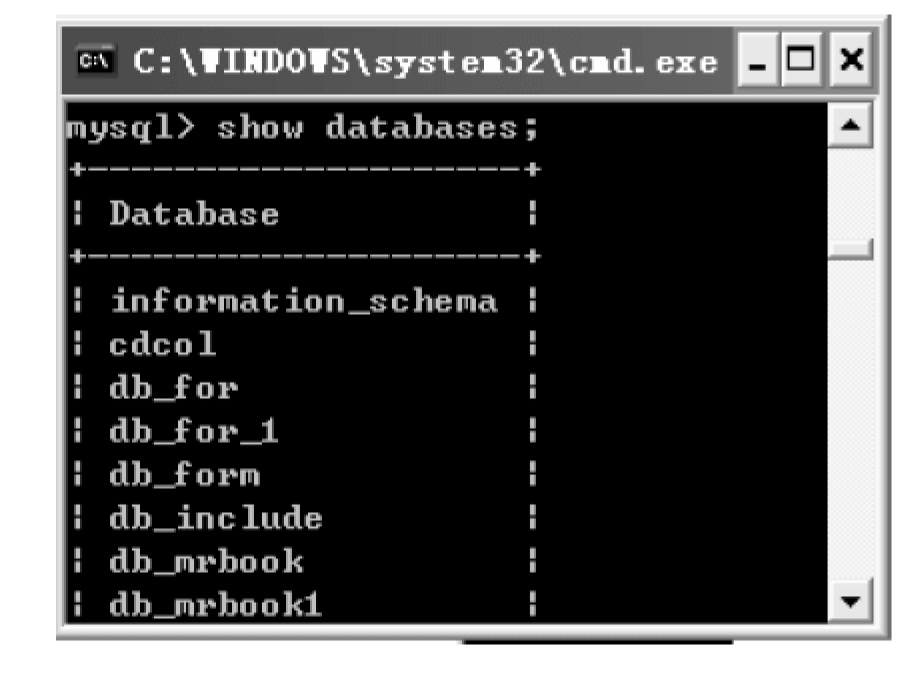


图 5.9　数据库 db_mrbook 存在于数据库列表中

（4）断开数据库连接，关闭 MySQL 服务。

技术要点

（1）创建数据库函数。

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name;
```

CREATE DATABASE 用于创建数据库，并进行命名。参数 db_name 表示需要创建的数据库名称。

（2）获取数据库列表。

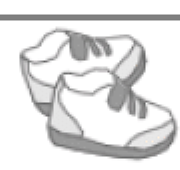
```
SHOW {DATABASES | SCHEMAS} [LIKE 'pattern'];
```

SHOW DATABASES 可以列举数据库名称。

实例 082　选择 PHP 图书数据库

实例说明

创建数据库的目的是向数据库中存储同类型或者同一个项目下的所有数据表。用户创建数据库后，还需要通过命令选择指定的数据库，否则计算机无法自动识别创建的数据表属于哪个数据库，还会出现错误提示，如图 5.10 所示。





本实例通过数据库函数“USE db_name”实现选择 PHP 图书数据库。

实现过程

具体步骤如下：

（1）在命令提示符窗口下，通过命令启动 MySQL 服务，并连接 MySQL。创建数据库并将此数据库命名为 db_mrbook。

（2）通过“show databases;”语句，查看在 MySQL 服务器主机上列举的数据库名称是否包含用户指定的数据库。

（3）输入命令“use db_mrbook;”，将数据库 use db_mrbook 作为当前默认数据库。按 Enter 键，命令窗口将输出如图 5.11 所示的内容。

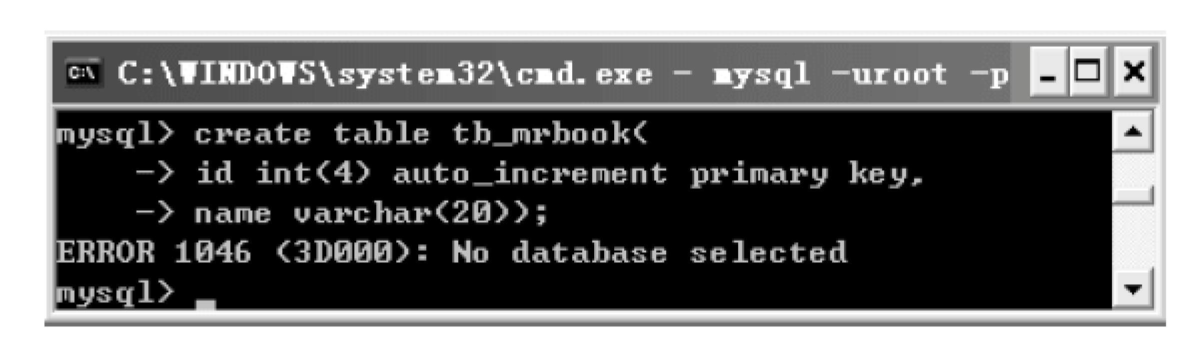


图 5.10　错误提示



图 5.11　选择图书数据库

（4）关闭数据库连接，关闭 MySQL 服务。

技术要点

USE db_name 语句可以通过 MySQL 把 db_name 数据库作为默认（当前）数据库，并应用于后续语句。该数据库保持为默认数据库，直到语段的结尾，或者直到发布一个不同的 USE 语句。语法格式如下：

```
USE db_name;
```

参数 db_name 表示需要使用的数据库名称，该名称必须是合法的。

实例 083　删除 PHP 图书数据库

实例说明

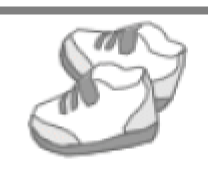
如果某个数据库失去了存在的价值，那么为了节省系统硬盘资源，提高 MySQL 的运行效率，建议从硬盘上删除此数据库。删除数据库可以通过打开 MySQL 安装目录，进入 data 文件夹，删除指定的数据库文件夹来完成，还可以通过在命令提示符下输入删除命令删除数据库。本实例讲解在命令提示符下输入命令，实现删除 PHP 图书数据库。

实现过程

具体步骤如下：

（1）进入命令提示符窗口，启动 MySQL 服务，连接 MySQL 服务器。

（2）通过命令语句“show databases;”，查看在 MySQL 服务器主机上列举的数据库名





称是否存在数据库“db_mrbook”。

（3）使用命令语句“use db_mrbook;”，为数据库“db_mrbook”做标记，使数据库“db_mrbook”为当前默认数据库，如图 5.12 所示。

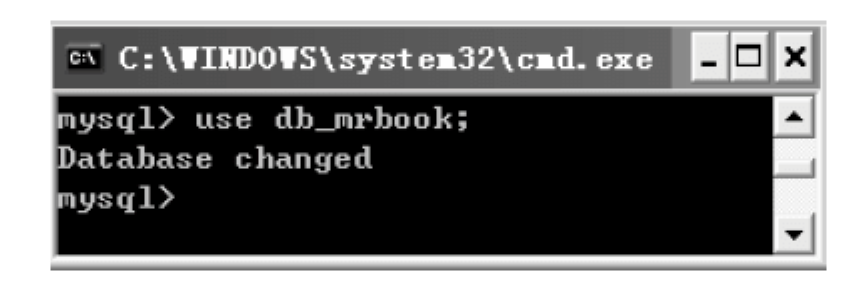


图 5.12　选择图书数据库

（4）在命令行输入“drop database db_name;”语句，删除数据库的同时也删除数据库中的所有表。所以笔者建议在命令行输入命令删除数据库时，最好清查数据库中的所有表名，以防止意外删除运行程序中的作用表，而造成不必要的损失。查看数据库中所有表名，可以通过命令语句“show tables;”实现。“show tables;”语句获取指定数据库中所有的表，如图 5.13 所示。

（5）使用命令行语句删除 PHP 图书数据库“db_mrbook”，如图 5.14 所示。

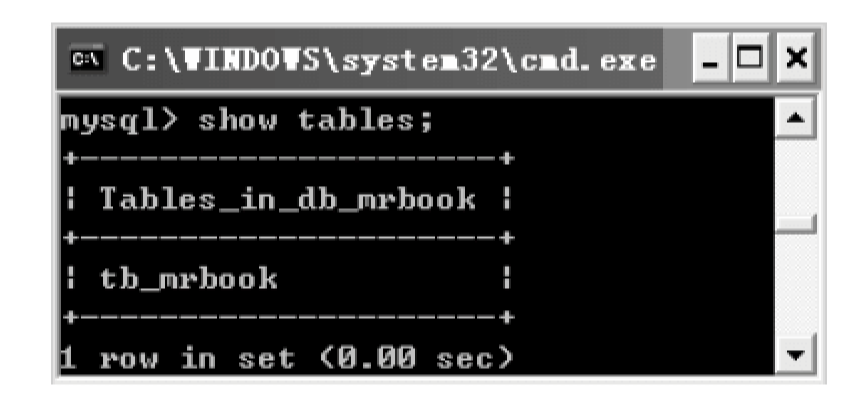


图 5.13　显示图书数据库中的数据表



图 5.14　删除图书数据库

（6）断开与 MySQL 服务器的连接，关闭 MySQL 服务。

技术要点

DROP DATABASE 语句用于删除数据库中的所用表格和数据库。使用此语句时要非常小心，如果要使用 DROP DATABASE，就必须获得数据库 DROP 权限。其语法格式如下：

```
DROP {DATABASE | SCHEMA} [IF EXISTS] db_name;
```

实例 084　在 PHP 图书数据库中创建图书信息表

实例说明

数据库的主要作用是保存数据表，而数据表的作用是保存数据信息。本实例通过“CREATE TABLE”语句实现在 PHP 图书数据库中创建图书信息表，如图 5.15 所示。

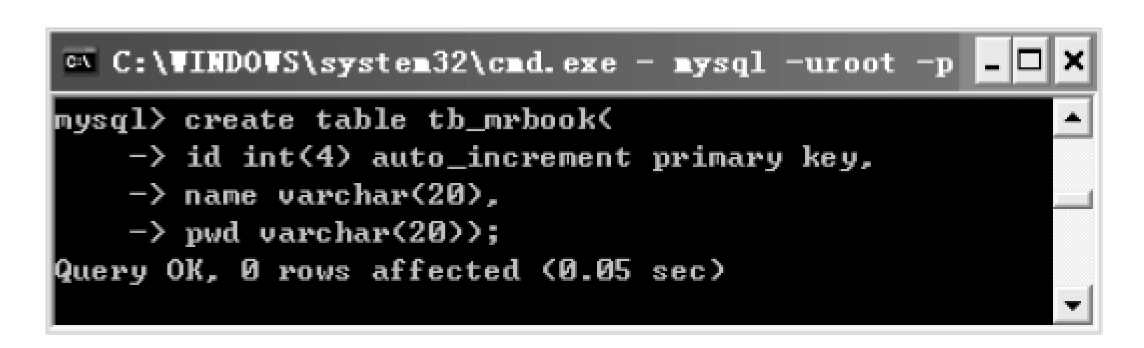


图 5.15　创建图书信息表



实现过程





（1）启动 MySQL 服务，建立 MySQL 服务器连接。

（2）在命令提示符窗口下通过命令语句“create database db_name;”创建名称为“db_mrbook”的数据库。

（3）使用“use db_name;”语句设定数据库标记。

（4）通过“create table tb_name();”语句创建数据表，代码如下：

```
mysql> use db_mrbook;
Database changed
mysql> create table tb_mrbook(
    -> id int(4) auto_increment prima
    -> name varchar(20) NOT NULL,
    -> pwd varchar(20) NOT NULL);
Query OK, 0 rows affected (0.05 sec)
```

（5）断开 MySQL 服务器连接，结束 MySQL 服务。

技术要点

创建数据表使用“CREATE TABLE”语句，其语法格式如下：

```
CREATE TABLE table_name
(create_definition,….)
[table_opotions]
[select_statement]
```

参数说明：

☑ table_name：要创建的数据表名。

☑ create_definition：这是表的列属性部分。MySQL 要求在创建表的时候，表要至少包含一列。

☑ table_options：表的一些特性参数。

☑ select_statement：SELECT 语句描述部分，用它可以快速地创建表。

下面介绍列属性部分，每一列定义的具体格式如下：

```
col_name type[NOT NULL | NULL][DEFAULT default_value][AUTO_INCREMENT][PRIMARY KEY]
```

参数说明：

☑ col_name：表示字段名。

☑ type：表示字段类型。

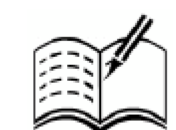
☑ NOT NULL 或者 NULL：指出该列是否允许是空值。所谓的空值是“不知道”或“无意义”的值，但是数据“0”和空格都不是空值，系统一般默认允许为空值，所以当不允许为空值时，必须使用 NOT NULL。

☑ DEFAULT default_value：表示默认值。

☑ AUTO_INCREMENT：表示是否为自动编号，每个表只能有一个 AUTO_INCREMENT 列，并且必须被索引。







☑ PRIMARY KEY：表示是否为主键。它是一个唯一的 KEY，还有一个额外的约束，即所有键列必须被定义为 NOT NULL。在 MySQL 中，该列被命名为 PRIMARY。一个表只能有一个 PRIMARY KEY。如表中没有一个 PRIMARY KEY，而某些应用程序要 PRIMARY KEY，MySQL 将返回第一个没有任何 NULL 列的 UNIQUE 键，作为 PRIMARY KEY。一个 PRIMARY KEY 可以是一个多列索引，但是不能在一个列规格说明中使用 PRIMARY KEY 键属性来创建一个多列索引。这样做将仅仅标记单个列作为主键，必须使用 PRIMARY KEY(index_col_name,…)句法。如果 PRIMARY KEY 或 UNIQUE 键只由一个列组成，并且列类型是整型，则可以用_rowid 引用它。

以上是创建一个数据表的一些基础知识，它看起来十分复杂，但在实际的应用中使用最基本的格式创建数据表即可，具体格式如下。

```
create table table_name(列名 1 属性，列名 2 属性…)
```

实例 085　查看图书信息表

实例说明

查看信息表分两种形式：第一种是查看信息表结构，第二种是查看信息表数据。查看信息表数据信息，需要使用“select”查询语句。本实例通过 show columns 命令查看信息表结构，运行结果如图 5.16 所示。

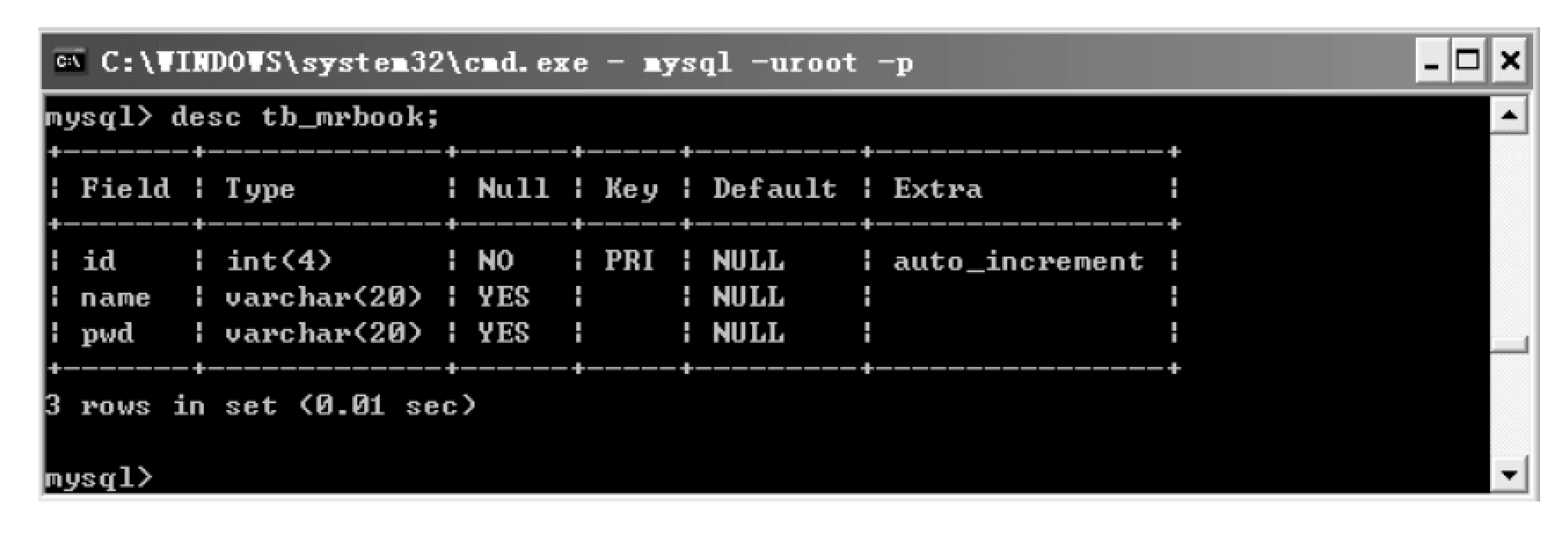


图 5.16　查看信息表结构图

实现过程

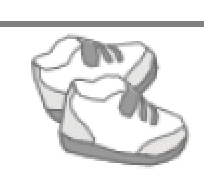
具体步骤如下：

（1）启动 MySQL 服务，在命令提示符窗口输入命令，建立 MySQL 连接。通过“create database db_name;”和“create table tb_name;”语句创建数据库和数据表。

（2）通过命令语句“show columns from tb_name;”查看表结构，其代码如下：

```
mysql> show columns from tb_mrbook;
```

（3）通过“exit”和“net stop mysql;”语句断开数据库服务器并关闭 MySQL 服务。



技术要点

查看数据表结构使用 show columns 命令，其语法如下：

```
show [full] columns   from  数据表名  [from  数据库名];
```

或写成

```
show   [full] columns   from  数据表名.数据库名;
```



实例 086　修改图书信息表

实例说明

程序员在建立数据表时，难免在表的结构上出现错误，例如，将“Date”类型数据设置为“int”类型，这时就需要更改表的结构。本实例通过增加表“tb_mrbook”中的“last_date”字段，并设置数据类型为“Date”实现修改信息表，运行结果如图 5.17 所示。

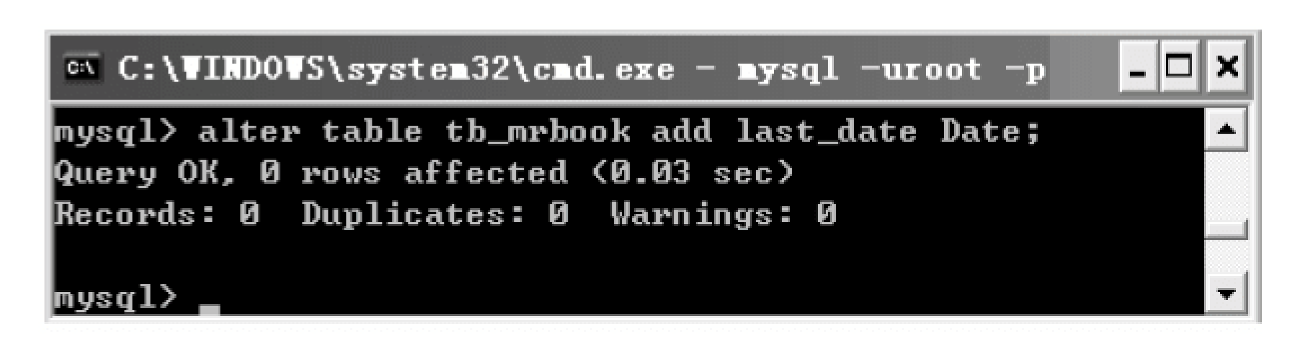


图 5.17　修改图书信息表

实现过程

具体步骤如下：

（1）启动 MySQL 服务，在命令提示符窗口下，建立 MySQL 连接。通过“create database db_name;”和“create table tb_name;”语句创建数据库和数据表。

（2）使用命令语句“show columns from tb_name;”查看表结构。

（3）在命令提示符下输入如下命令：

```
mysql> alter table tb_mrbook add last_date Date;
```

按 Enter 键，运行结果如图 5.17 所示。

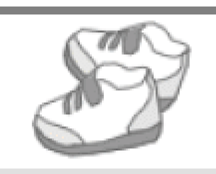
（4）通过“exit”和“net stop mysql;”语句断开数据库服务器并关闭 MySQL 服务。

技术要点

修改表结构采用“alter table”命令。修改表结构是指增加或者删除字段、修改字段名称或者字段类型、设置取消主键外键、设置取消索引以及修改表的注释等。其语法如下：

```
alter [IGNORE] table  数据表名  alter_spec[,alter_spec]…;
```

注意，当指定 IGNORE 时，如果出现重复的行，则只执行一行，其他重复的行被删除。其中，alter_spec 子句定义要修改的内容，其语法如下：





```
alter_specification:
    ADD [COLUMN] create_definition [FIRST | AFTER column_name ]          --添加新字段
  | ADD INDEX [index_name] (index_col_name,...)                          --添加索引名称
  | ADD PRIMARY KEY (index_col_name,...)                                 --添加主键名称
  | ADD UNIQUE [index_name] (index_col_name,...)                         --添加唯一索引
  | ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}         --修改字段名称
  | CHANGE [COLUMN] old_col_name create_definition                       --修改字段类型
  | MODIFY [COLUMN] create_definition                                    --修改子句定义字段
  | DROP [COLUMN] col_name                                               --删除字段名称
  | DROP PRIMARY KEY                                                     --删除主键名称
  | DROP INDEX index_name                                                --删除索引名称
  | RENAME [AS] new_tbl_name                                             --更改表名
  | table_options
```

实例 087　重命名图书信息表

实例说明

MySQL 数据表是不允许重名的，所以在同一个数据库中绝对不会出现两个名称相同的表，但是一般情况下，程序员希望自己创建的数据表能更贴近自己的项目，所以对数据表重命名有时是不可避免的。本实例通过“rename table”命令实现对图书信息表重命名，运行结果如图 5.18 所示。

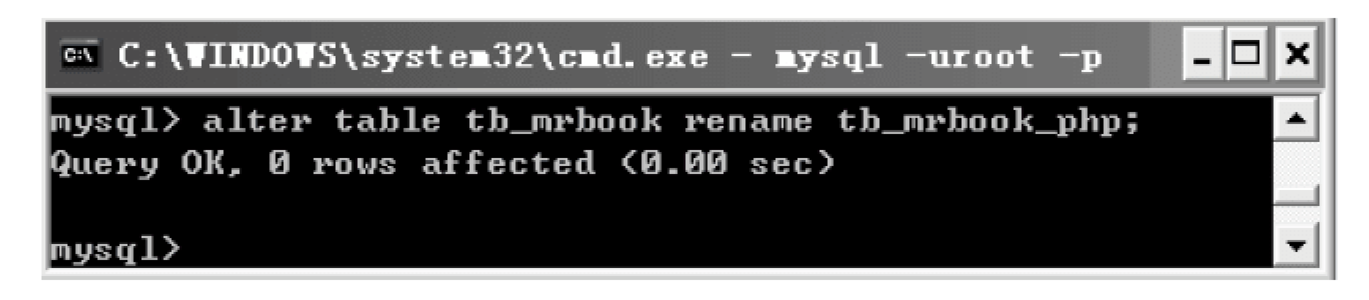


图 5.18　重命名图书信息表

实现过程

具体步骤如下：

（1）在命令提示符窗口下，启动 MySQL 服务，建立 MySQL 连接，创建数据库并建立数据表。

（2）使用命令语句“show tables;”显示当前数据库中的所有表。

（3）在命令提示符下输入如下命令：

```
mysql> alter table tb_mrbook rename tb_mrbook_php;
```

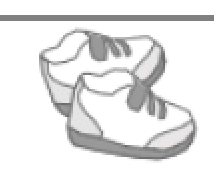
按 Enter 键，完成数据表的重命名操作，运行结果如图 5.18 所示。

（4）断开 MySQL 服务器连接，关闭 MySQL 服务。

技术要点

重命名数据表采用 rename table 命令，语法格式如下：

```
rename table 数据表名 1 To 数据表名 2;
```



或者

```
alter table 数据表名 1 rename 数据表名 2;
```



Note

实例 088　删除图书信息表

实例说明

“drop database db_name”语句删除数据库的同时会连带删除数据库中的所有数据表，如果只想删除数据库中的某个表，而保留其他所有表，显然“drop database db_ name”语句已经不再适用，这时可以使用“drop table tb_name;”语句，运行结果如图 5.19 所示。

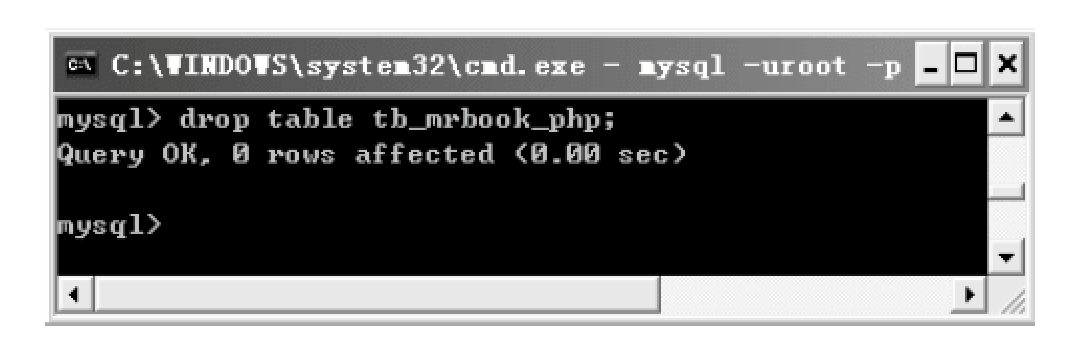


图 5.19　删除图书信息表

实现过程

具体步骤如下：

（1）在命令提示符窗口下，启动 MySQL 服务，通过输入命令连接到 MySQL 服务器。

（2）使用命令语句“show tables;”显示当前数据库中的所有表。

（3）在命令提示符下输入如下命令，删除指定数据表。

```
drop table tb_mrbook;
```

按 Enter 键，运行结果如图 5.19 所示。

（4）断开 MySQL 服务器连接，关闭 MySQL 服务。

技术要点

删除数据表的操作很简单，同删除数据库的操作类似，使用 drop table 命令就可以实现。其语法格式如下：

```
drop table 数据表名;
```

实例 089　向图书信息表中添加数据

实例说明

建立数据库是为了把不同类型或者不同要求的数据分类存储在不同的数据表中，创建数据表是为了将相同类型或者相同要求的数据存储在同一表中。本实例讲解通过 SQL 语句向图书信息表中添加数据，运行结果如图 5.20 所示。





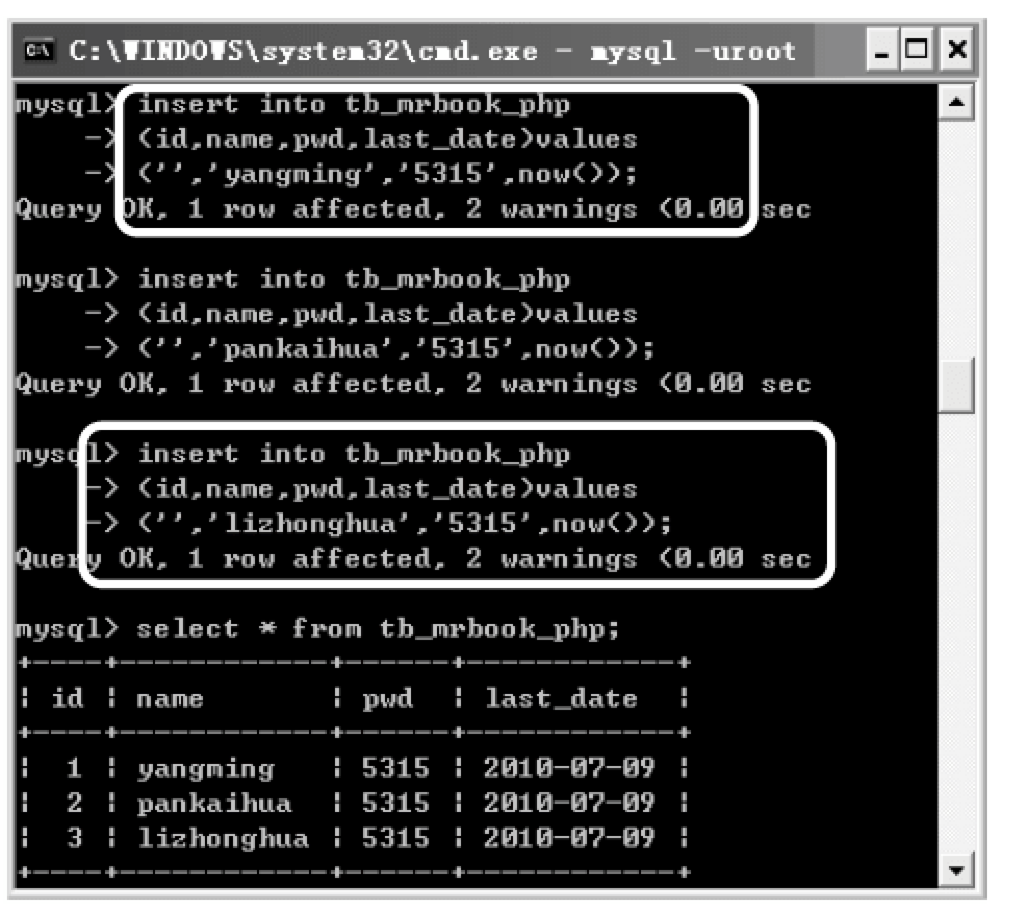




图 5.20　向图书信息表中添加数据

实现过程

具体步骤如下：

（1）在命令提示符窗口下，启动 MySQL 服务，通过命令行语句连接 MySQL 服务器，创建数据库并建立数据表。

（2）编辑 Insert 语句，执行数据的添加操作，其代码如下：

```
insert into tb_mrbook_php(id,name,pwd,last_date)values ('','lizhonghua','5315',now());
```

（3）当提示符窗口出现如图 5.21 所示内容时，表示数据添加成功。

```
选定 C:\WINDOWS\system32\cmd.exe - mysql -uroot -p
mysql> insert into tb_mrbook_php
    -> (id,name,pwd,last_date)values
    -> ('','pankaihua','5315',now());
Query OK, 1 row affected, 2 warnings (0.00 sec)
```

图 5.21　数据添加成功

（4）通过 SELECT 语句查询数据表中的内容，显示结果如图 5.20 所示。

（5）断开 MySQL 服务器连接，关闭 MySQL 服务。

技术要点

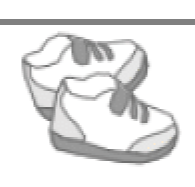
建立一个空的数据表时，首先要想到的就是如何向数据表中添加数据。这项操作可以通过 INSERT 命令来实现。Insert 语句的语法如下：

```
insert  into 数据表名(column_name,column_name2, … ) values (value1, value2, … );
```

实例 090　修改图书信息表中的数据

实例说明

用户向数据表中插入数据信息时，在手动输入的过程中很有可能将信息输入错误。这





个时候就需要 SQL 语句对错误的数据进行修改。本实例通过修改语句，将图书信息表中操作数据表的时间进行修改，运行结果如图 5.22 所示。

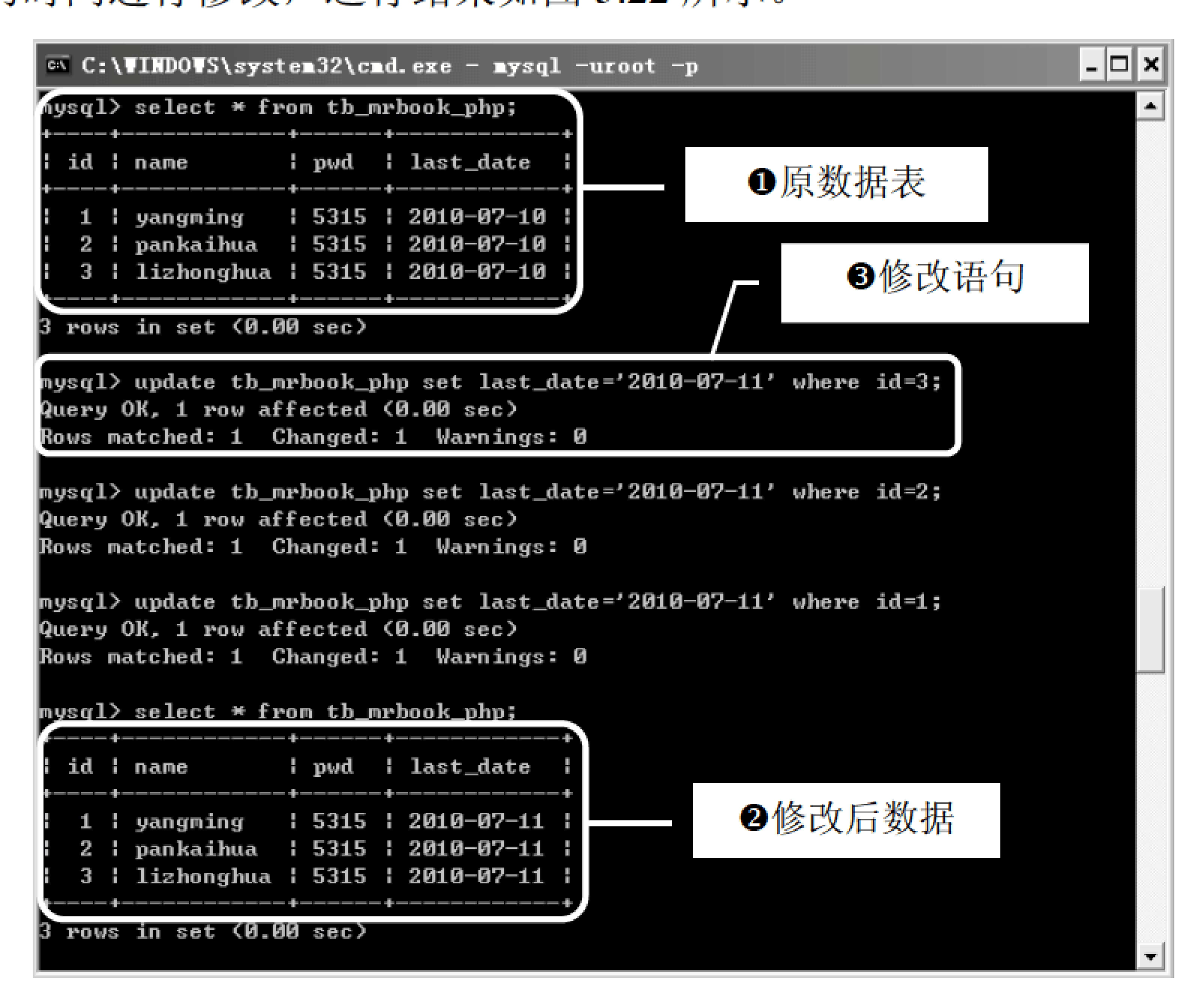


图 5.22　修改图书信息表中的数据

实现过程

具体步骤如下：

（1）在命令提示符窗口下，启动 MySQL 服务，通过命令行语句连接 MySQL 服务器，创建数据库并建立数据表。

（2）使用“select”查询语句查询图书信息表，如图 5.22 中❶原数据表所示。

（3）使用“update”更新语句，更新数据表中的数据，其代码如下：

```
update tb_mrbook_php set last_date='2010-07-11' where id=1;
```

（4）当提示符窗口出现如图 5.23 所示内容时，表示数据更新成功。

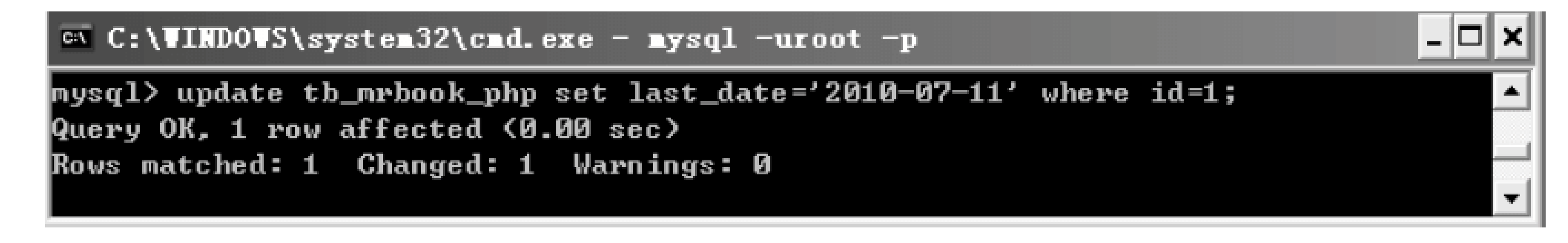


图 5.23　修改成功提示

（5）通过 SELECT 语句查询数据表中内容，显示结果如图 5.23 中❷修改后数据所示。

（6）断开 MySQL 服务器连接，关闭 MySQL 服务。

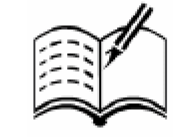
技术要点

执行修改操作应用的是 update 命令，该语句的语法格式如下：

```
update 数据表名 set column_name = new_value1,column_name2 = new_value2, …where condition
```



其中，set 子句指出要修改的列及其给定的值；where 子句是可选的，如果给出该子句，将指定记录中哪一行应该被更新，否则，所有的记录行都将被更新。



实例 091　删除图书信息表中所有数据

实例说明

删除数据表中的所有数据在实际开发中是不多见的，删除数据表中的所有数据意味着清空数据表中的所有信息，如果是操作上的失误造成数据表中所有信息丢失，对于程序员来说是灾难性的。所以对数据表中信息的删除操作是需要十分谨慎的。本实例通过两种方法实现删除图书信息表中所有数据，运行结果如图 5.24 和图 5.25 所示。

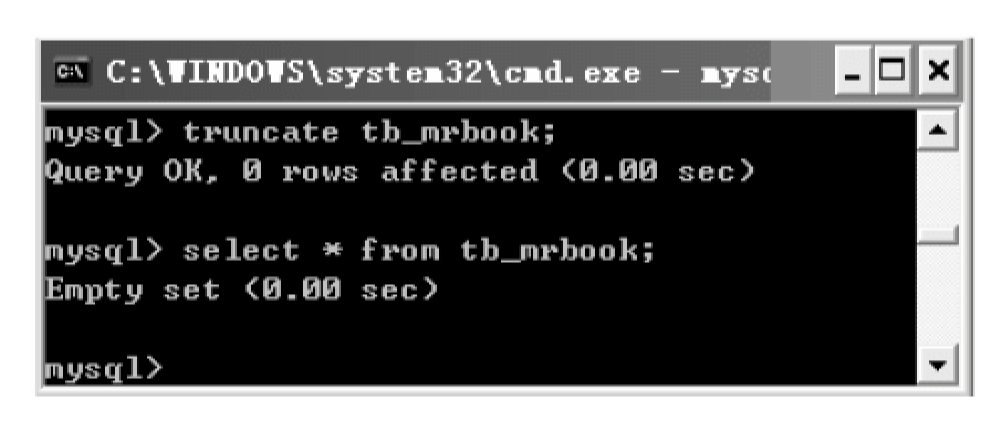


图 5.24　清空数据库

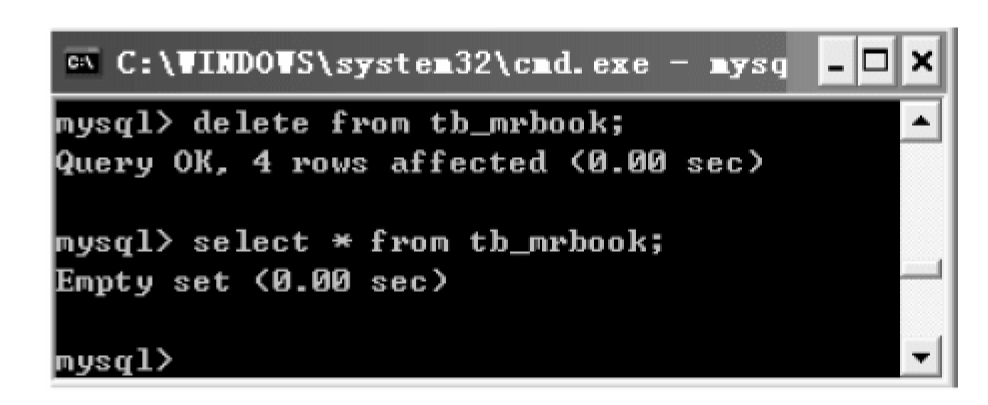


图 5.25　删除所有数据

实现过程

具体步骤如下：

（1）启动 MySQL 服务，通过命令提示符窗口建立 MySQL 连接，新建数据库和数据表，将数据表命名为“tb_mrbook”，利用 INSERT 插入语句向数据表中插入数据，并通过 SELECT 语句查询数据表。

（2）利用 TRUNCATE 语句，清空数据表中所有数据，其代码如下：

```
truncate tb_mrbook;
```

运行结果如图 5.24 所示。

（3）利用 DELETE 语句删除所有数据，其代码如下：

```
delete from tb_mrbook;
```

运行结果如图 5.25 所示。

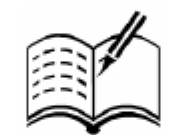
技术要点

（1）删除数据信息使用 DELETE 语句，其语法如下：

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name
    [WHERE where_definition]
    [ORDER BY ...]
    [LIMIT row_count]
```

该语句在执行过程中，删除 table_name 表中的记录，如果没有指定 WHERE 条件，将





删除所有的记录；如果指定 WHERE 条件，将按照指定的条件进行删除。参数 ORDER BY 表示按条件排序，参数 LIMIT row_count 表示控制显示条数。

（2）清空数据表信息使用 TRUNCATE 语句，其语法如下：

```
TRUNCATE [TABLE] tb_name;
```

用于完全清空数据表，从逻辑上说，该语句与用于删除所有行的 DELETE 语句等同。

实例 092　删除图书信息表中指定数据

实例说明

删除图书信息表中的所有数据或者说清空数据表的方法在实际编程中是很少见的，一般都是指定删除一条或几条数据。本实例通过 DELETE 语句的子句实现删除图书信息表中指定数据，运行结果如图 5.26 所示。

实现过程

具体步骤如下：

（1）启动 MySQL 服务，输入命令连接 MySQL，创建数据库和数据表，将数据表命名为 tb_mrbook 并通过 INSERT 语句向数据表里添加信息，通过 SELECT 语句查看表信息。

（2）通过 DELETE 语句删除 id 等于 5 的数据，运行结果如图 5.27 所示。



图 5.26　指定删除 id 等于 5 的数据

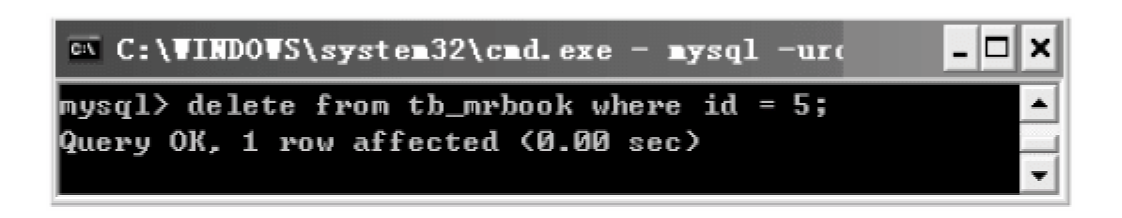


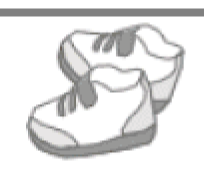
图 5.27　删除成功提示

技术要点

本实例的关键点是 WHERE 条件句的灵活运用。WHERE 子句是用来选取需要检索的记录。因为一个表通常会有数千条记录，在查询结果中，用户仅需其中的一部分记录，这时需要使用 WHERE 子句指定一系列的查询条件。下面是 WHERE 子句最简单的语法：

```
SELECT<字段列表>
FROM<表名>
WHERE<条件表达式>
```

为了满足多种不同的查询，WHERE 子句提供了丰富的搜索条件，下面将给出 WHERE 子句常用的比较运算符，如表 5.1 所示。



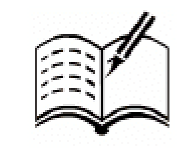


表 5.1 Where 子句的常用比较运算符

运 算 符	说 明	运 算 符	说 明
=	等于	<=	小于等于
>	大于	!>	不大于
<	小于	!<	不小于
>=	大于等于	<>或!=	不等于

实例 093 通过 phpMyAdmin 修改 MySQL 用户密码

实例说明

图形化管理工具 phpMyAdmin 的出现使 MySQL 数据库具有了如 Access 和 SQL Server 数据库一样简单而明了的操作界面，使程序员在设计 MySQL 数据库时变得简单。phpMyAdmin 几乎可以实现操作数据库的所有功能，本实例将演示如何通过 phpMyAdmin 修改 MySQL 用户密码。

实现过程

通过修改配置文件实现修改密码。

（1）将下载的 phpMyAdmin-x.zip 文件解压到 Apache 默认的根目录下，解压后的名称是 phpMyAdmin-2.x.x.x，其中的 2.x.x.x，是版本号。为了方便使用，可以将其重新命名为 phpMyAdmin。

（2）打开 phpMyAdmin 文件夹，找到配置文件 config.php，然后使用记事本或其他编辑工具打开该文件，找到该文件中的以下内容进行配置。

```
$cfg['PmaAbsoluteUri'] = 'http://127.0.0.1/phpMyAdmin';
```

以上用于设置 phpMyAdmin 的 URL，例如，http://127.0.0.1/phpMyAdmin。

```
$cfg['blowfish_secret'] = 'root';
```

更改上述语句中的用户密码，例如，“root”。

```
$cfg['Servers'][$i]['controluser’] = 'root';
$cfg['Servers'][$i]['controlpass’] = 'root';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'root';
```

（3）完成以上配置后保存该文件。

通过 phpMyAdmin 的界面进行密码修改。

（1）登录到 phpMyAdmin 图形化工具页面，单击“修改密码”按钮，如图 5.28 所示。

（2）根据页面提示，输入新密码并确认密码，如图 5.29 所示。

（3）单击“执行”按钮，修改 MySQL 数据库密码，如图 5.30 所示。

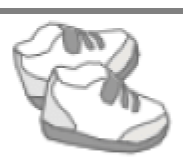








图 5.28　phpMyAdmin 图形化界面



图 5.29　修改 MySQL 数据库密码



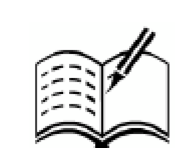
图 5.30　完成密码修改

技术要点

通过 phpMyAdmin 修改 MySQL 用户的密码可以通过两种方法实现。

（1）直接通过 phpMyAdmin 的界面进行更改。





（2）通过修改 phpMyAdmin 文件夹下的配置文件 config.php 实现更改。

实例 094　通过 phpMyAdmin 设置数据库、数据表编码

实例说明

将页面、程序文件、数据库与数据表设置成统一的编码格式可以使程序运行时不至于出现乱码。一般情况下，设置页面的编码格式由 HTML 中的 meta 标签实现，设置程序文件的编码格式由 mysql_query()实现，设置数据库与数据表的编码格式可以通过使用 phpMyAdmin 实现。

实现过程

具体步骤如下：

（1）登录到 phpMyAdmin 图形化工具页面，创建数据库名称，选择编码格式，如图 5.31 所示。

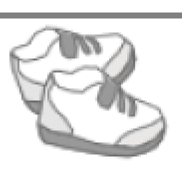


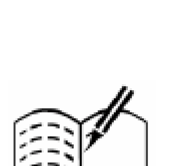
图 5.31　设置数据库编码格式

（2）创建数据表，定义数据表字段，并设置编码格式，如图 5.32 所示。

技术要点

通过 phpMyAdmin 设置数据库、数据表编码的操作方法十分简单。在创建数据库名称时，在“整理”下拉列表中选择需要设置的编码格式；在创建数据表时，同样将数据字段设置成统一的编码格式。





Note



图 5.32　设置字段编码格式

实例 095　phpMyAdmin 操作数据库

实例说明

phpMyAdmin 可以在 cmd 命令行下实现对数据库的操作，而且操作更简单、直观。本实例通过 phpMyAdmin 实现对数据库的创建和删除操作。

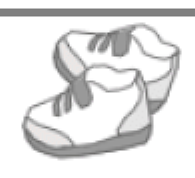
实现过程

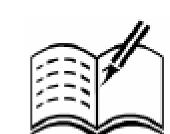
具体步骤如下：

（1）登录到 phpMyAdmin 图形化工具页面，根据提示编写新建数据库名称，并设置数据库的编码格式，如图 5.33 所示。



图 5.33　创建数据库





（2）单击“创建”按钮，创建数据库并自动选择创建的数据库为当前数据库。

（3）如果数据库已经没有任何作用，可以通过 phpMyAdmin 删除数据库，选择想要删除的数据库，单击导航栏中的“删除”按钮，根据提示决定是否删除，如图 5.34 所示。

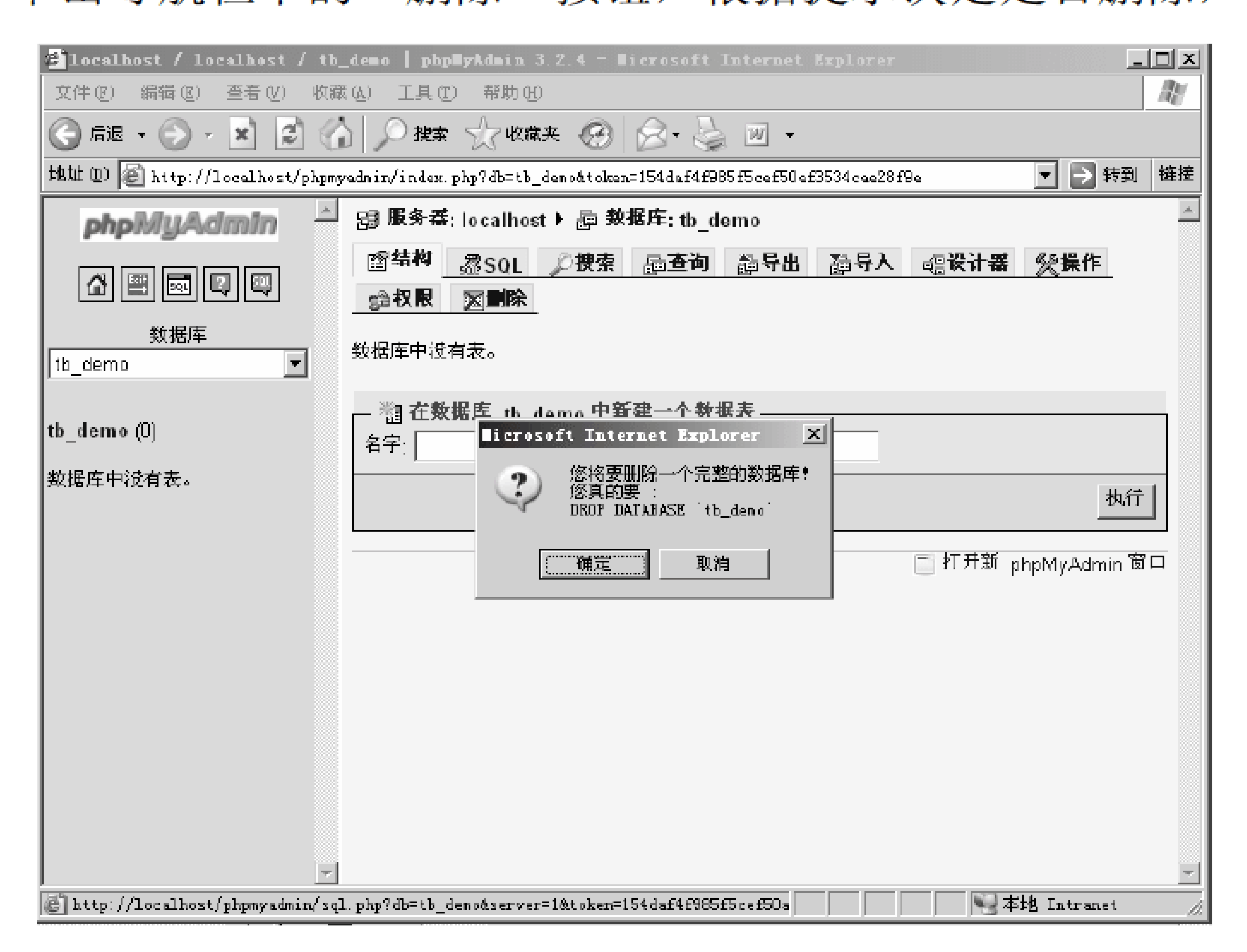


图 5.34　删除数据库

技术要点

切记在创建数据库时，一定要设置数据库的编码格式，而且一旦执行删除操作，数据库将不能再恢复。

实例 096　phpMyAdmin 操作数据表

实例说明

一般对数据表的操作包括创建和删除，下面介绍应用 phpMyAdmin 图形化工具实现数据表的创建和删除操作。

实现过程

具体步骤如下：

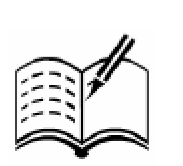
（1）登录到 phpMyAdmin 图形化工具页面，创建数据库，选择数据库编码格式，单击“创建”按钮，根据提示编写数据表名称，并填写想要创建的数据表的字段数目，如图 5.35 所示。

（2）进入编写数据库字段页，建立数据表结构，填写字段信息，其中包括字段名称、字段类型、字段长度、是否允许为空等，如图 5.36 所示。

（3）单击“保存”按钮，完成数据表的创建。

（4）删除数据表，首先选择即将删除的数据表名称，并进入数据表操作界面，单击





导航栏“删除”按钮，根据提示删除数据表，如图5.37所示。

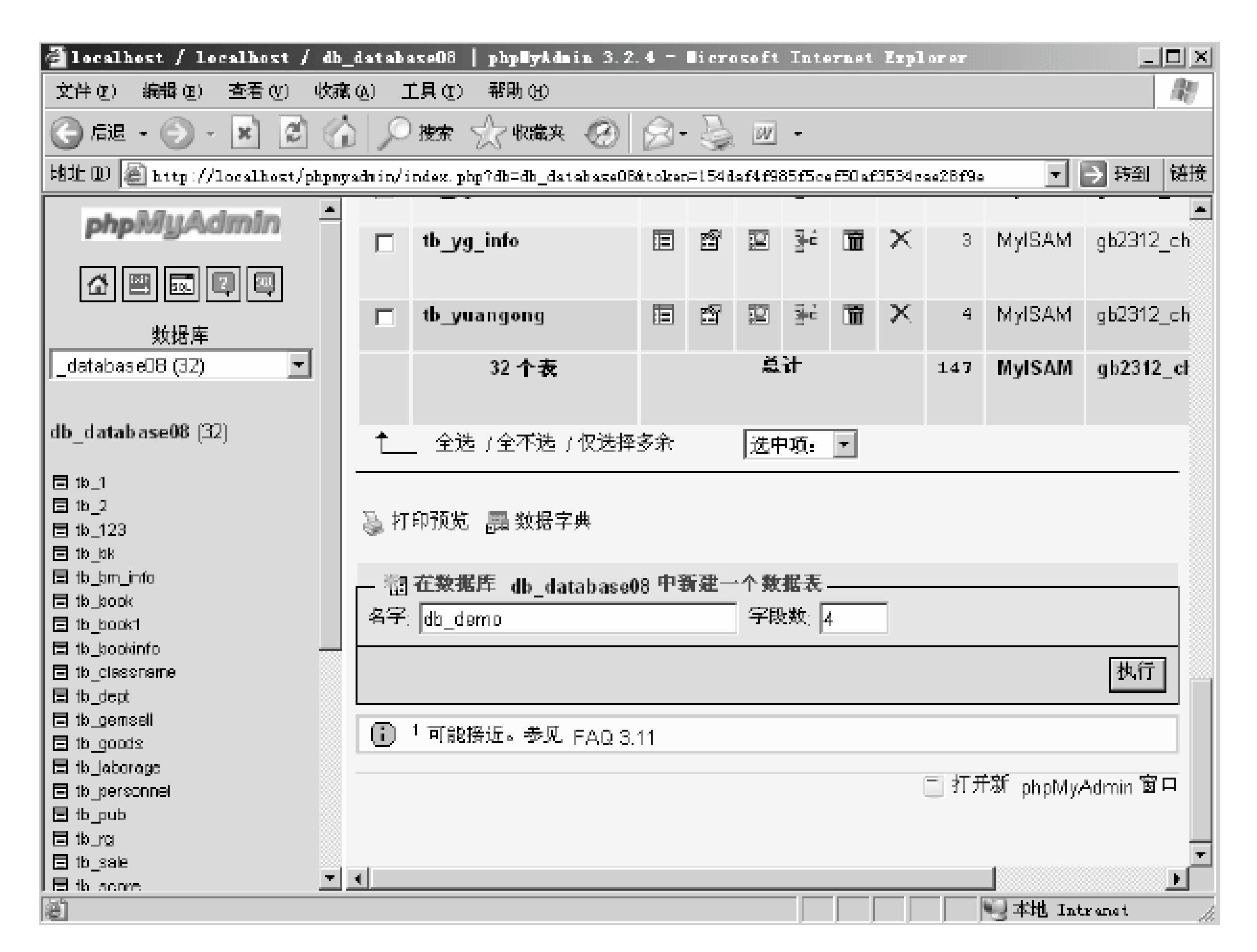


图5.35　创建数据表

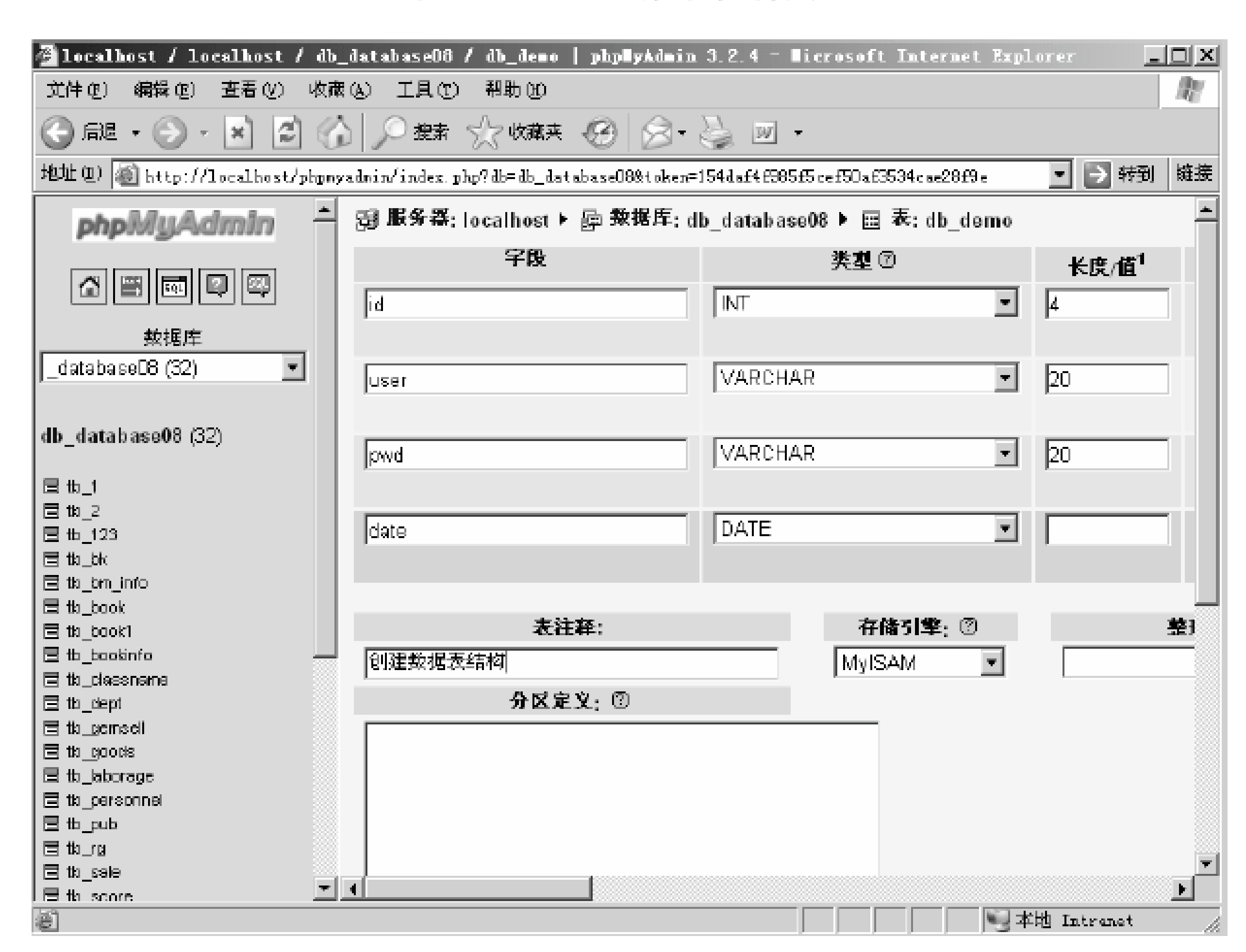


图5.36　创建数据表结构

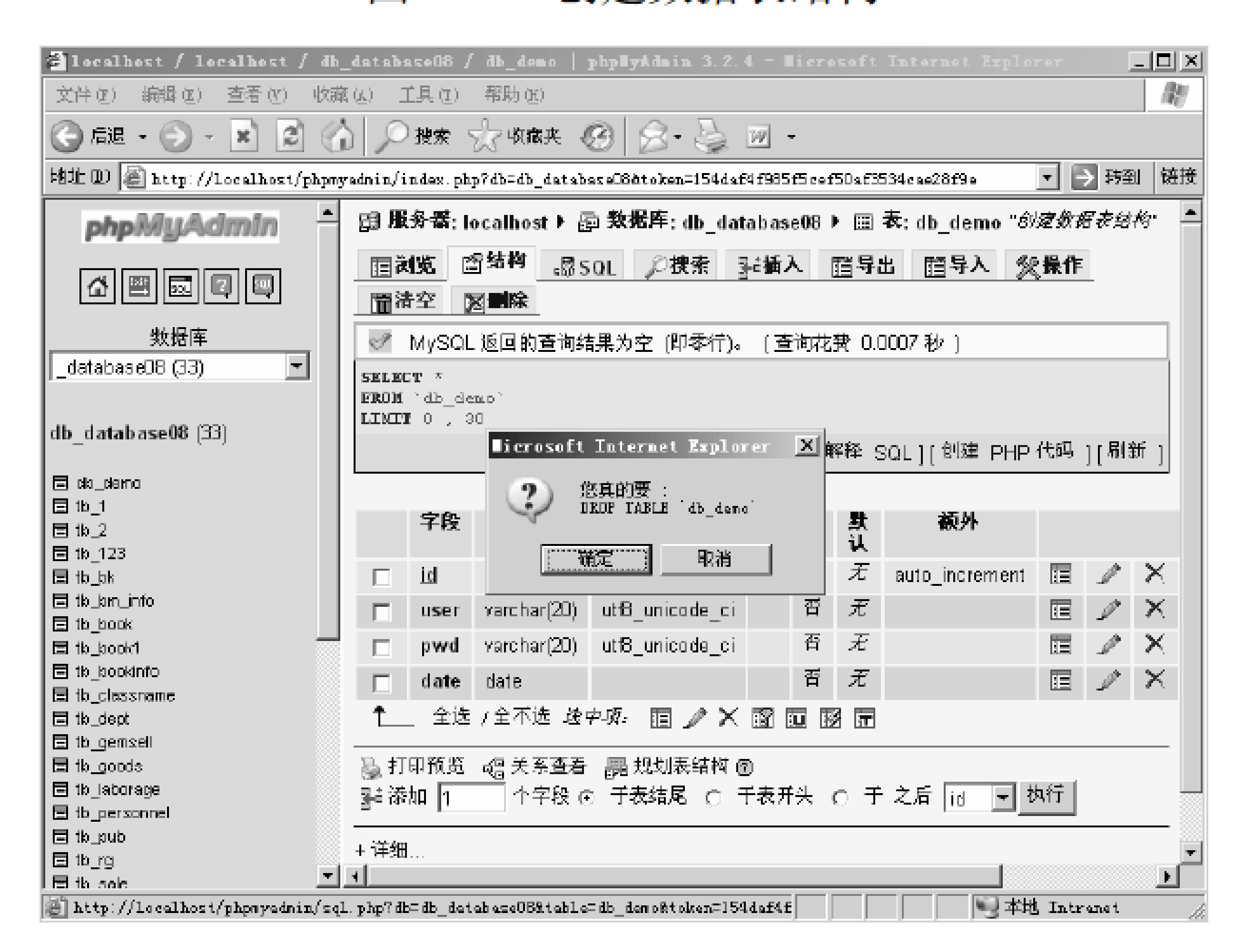
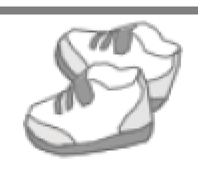
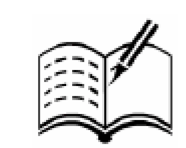


图5.37　删除数据表





技术要点

想要创建数据表首先要创建数据库，当数据库创建完成时自动选择已经创建好的数据库，根据提示编写数据表名称、创建数据表。而删除数据表要先选择想要删除的表，然后单击“删除”按钮即可完成。

实例 097　phpMyAdmin 操作数据

实例说明

对 MySQL 数据库数据的操作包括插入数据、删除数据、修改数据和查询数据等。本实例将介绍如何通过 phpMyAdmin 图形化管理工具完成上述的操作。

实现过程

具体步骤如下：

（1）查询数据信息：首先进入 tb_demo 数据表，默认情况下直接显示数据表的信息；如果刚刚进行完其他操作，则单击导航栏“浏览”按钮，显示数据表信息，如图 5.38 所示。

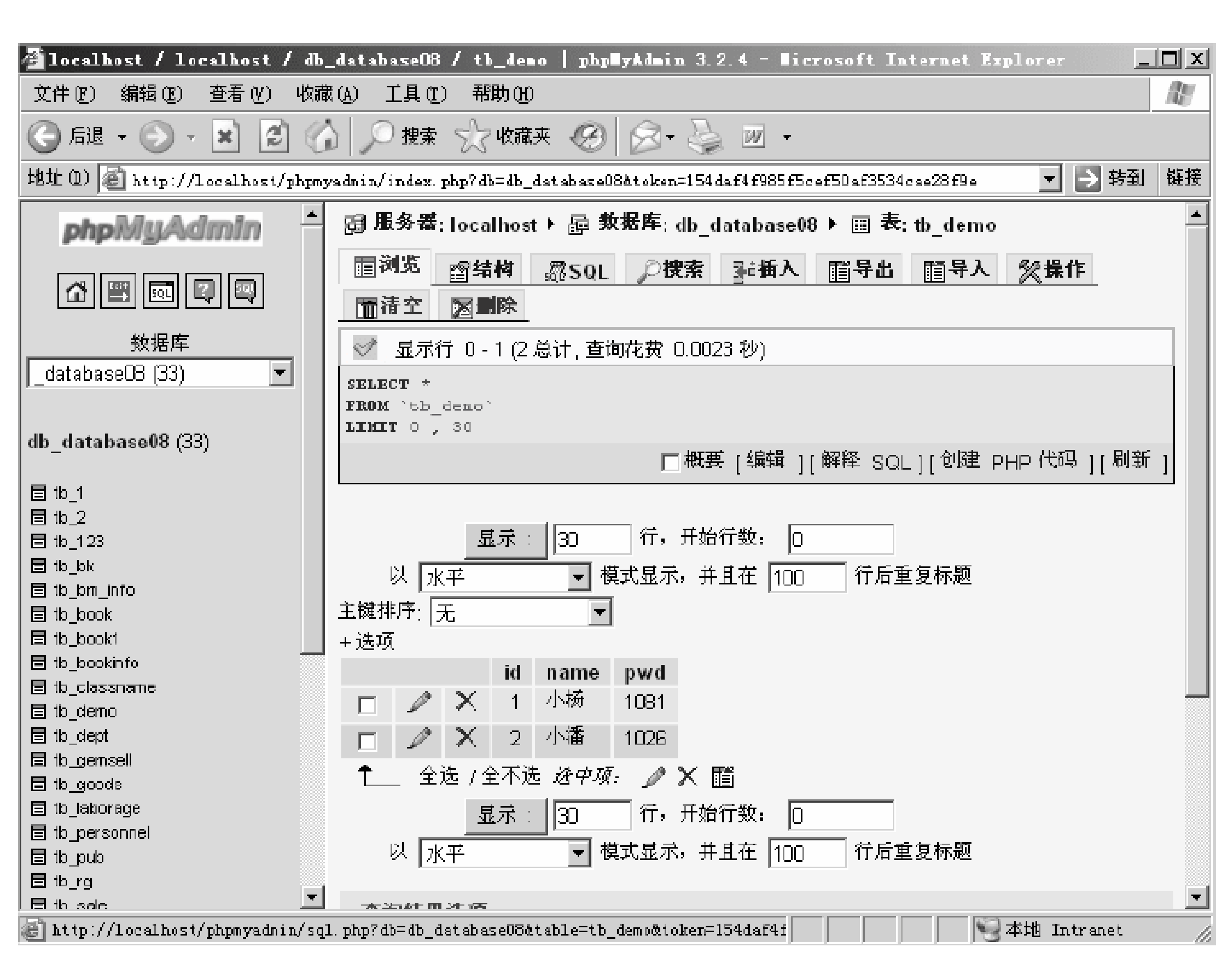
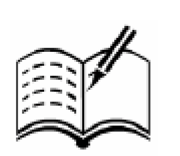


图 5.38　查询数据信息

（2）插入数据信息：单击导航栏“插入”按钮，进入插入信息页面，完成数据的添加，单击“执行”按钮实现插入数据，如图 5.39 所示。

（3）删除数据信息：首先返回到数据浏览页面，如果删除单条数据信息，直接单击该条数据的删除按钮，即可完成单条数据的删除；如果删除多条数据信息，可以选中数据信息前端的复选框，然后单击数据表信息下端的“选中项删除”按钮，完成多条信息删除；







如果想要删除数据表中的所有信息，可以单击导航栏的“清空”按钮，实现删除所有数据信息，如图 5.40 所示。

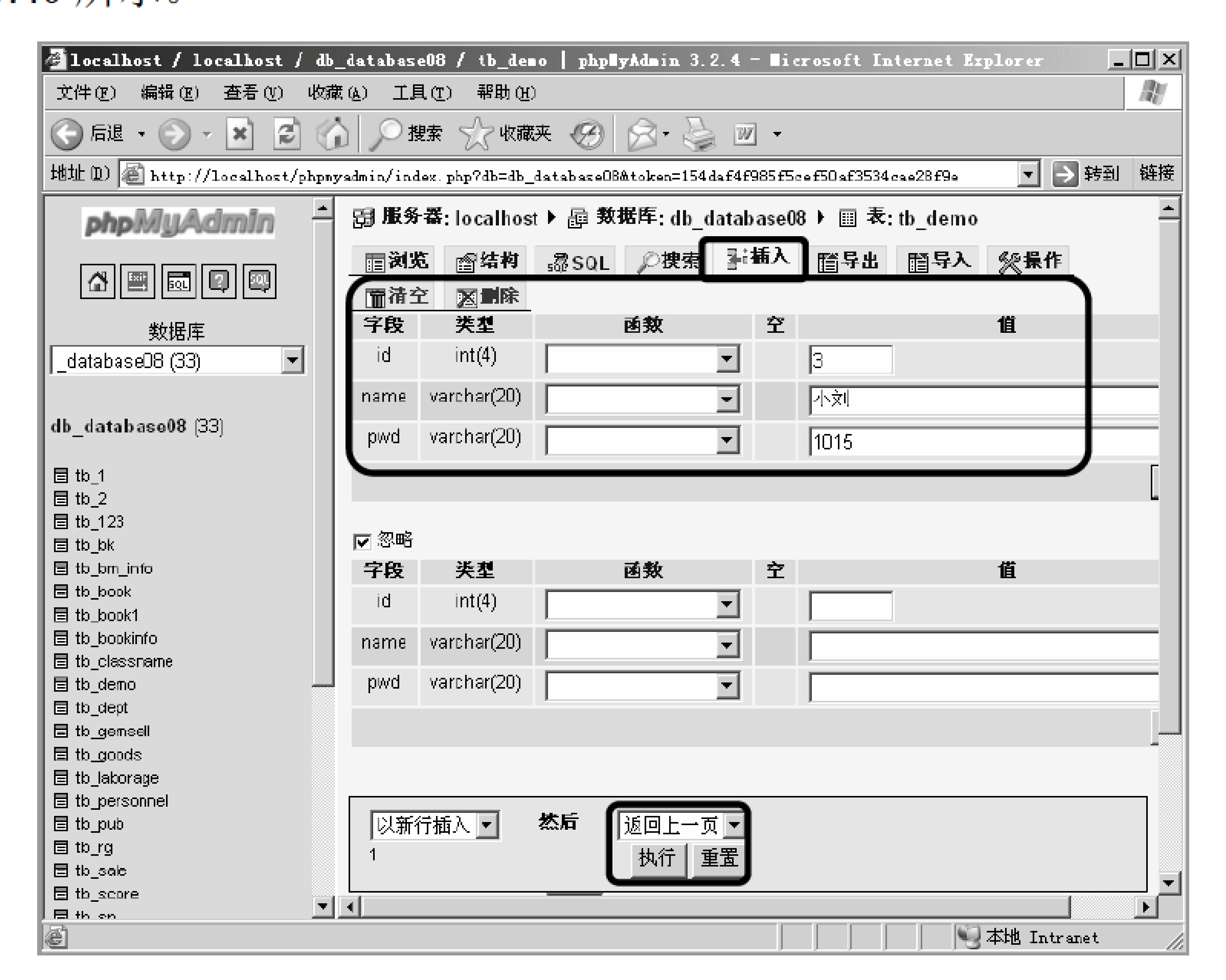


图 5.39　插入数据信息

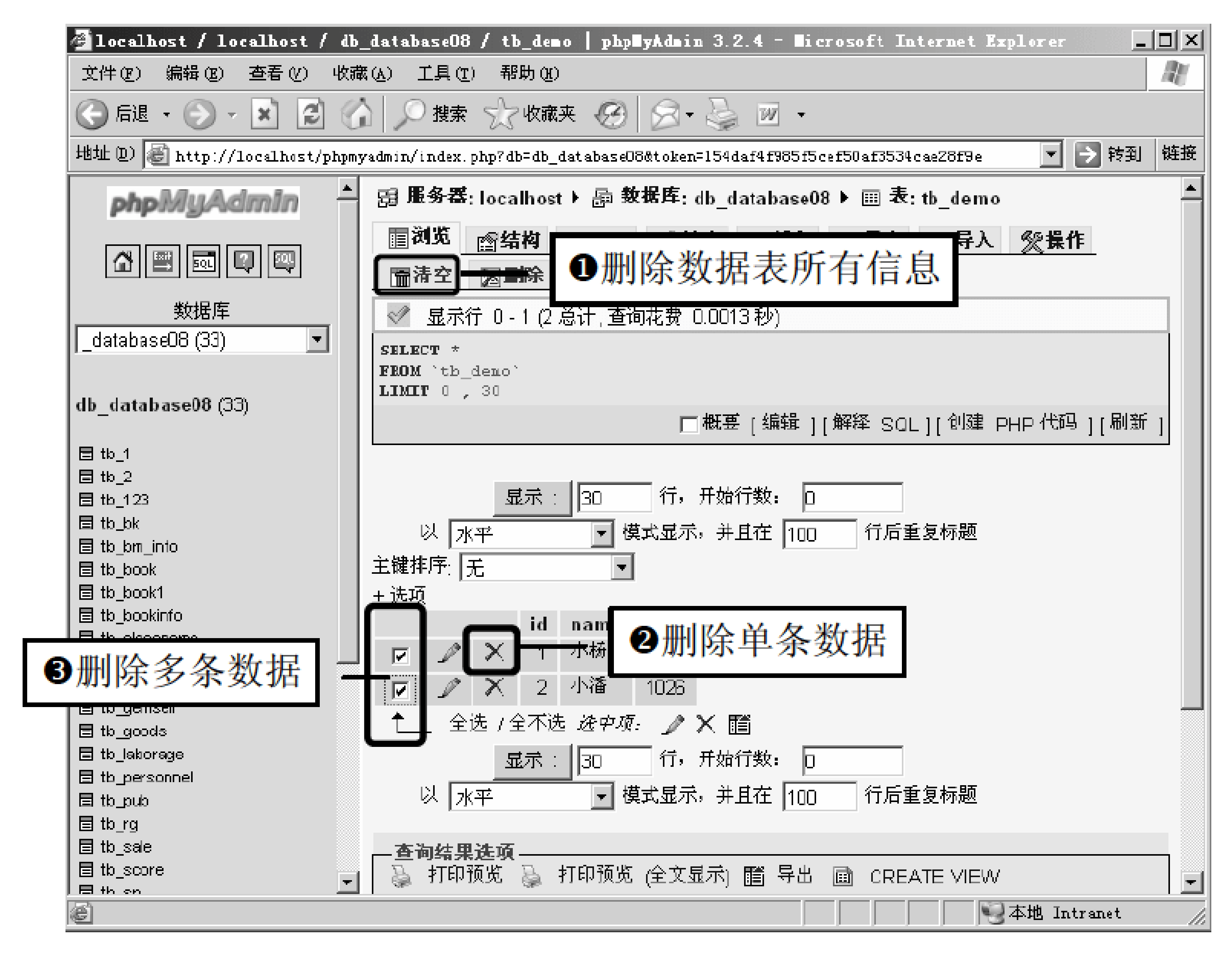
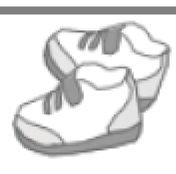


图 5.40　删除数据操作

（4）修改数据信息：修改数据信息与删除数据信息类似，可以分为修改单条数据信息和修改多条数据信息。修改单条数据信息直接单击该条数据的修改按钮，进入修改页面，实现修改。修改多条数据信息首先要选中复选框，然后单击“选中项修改”按钮，进入修改页面，完成修改，如图 5.41 和 5.42 所示。





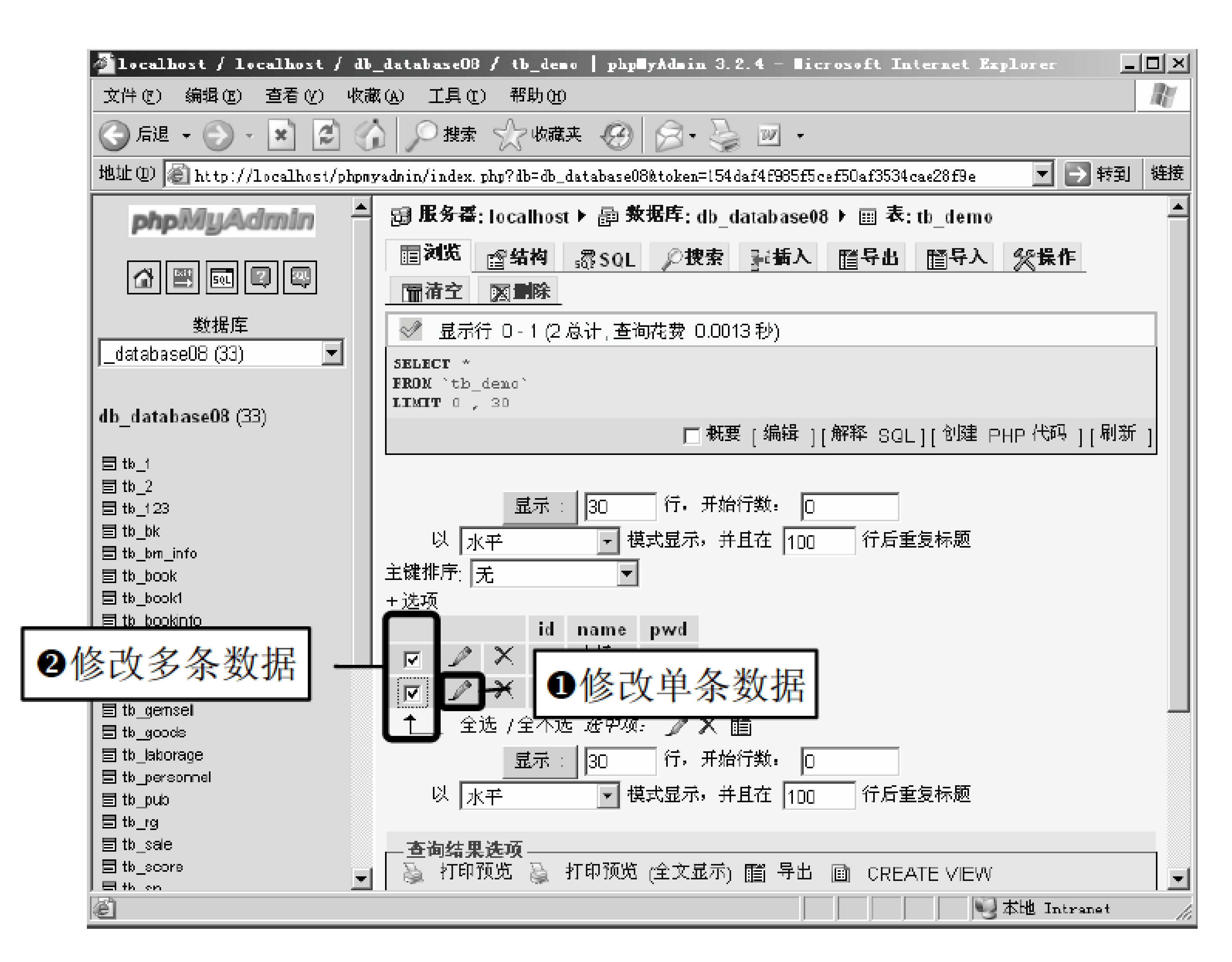


图 5.41　修改数据信息

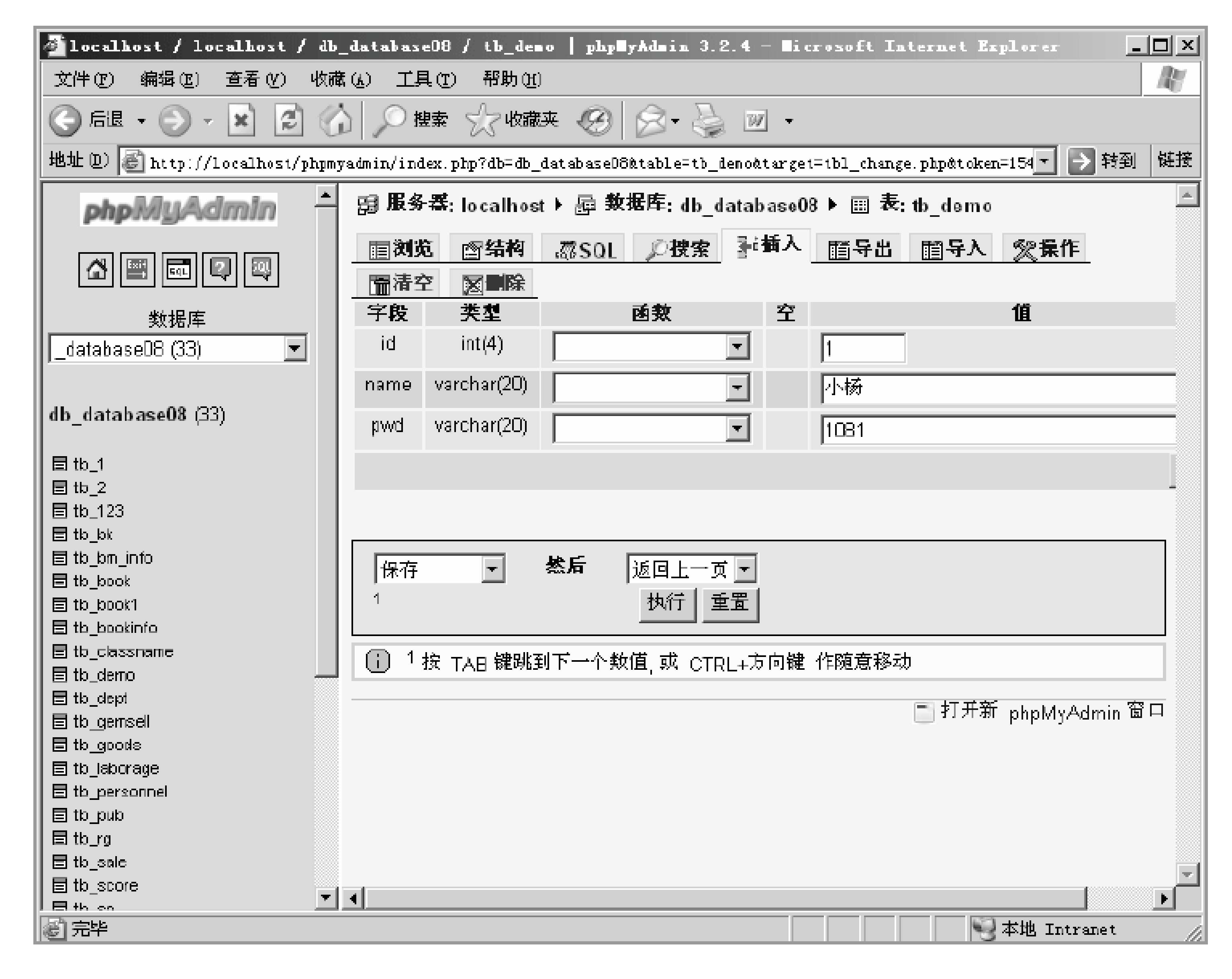


图 5.42　更改数据信息页

技术要点

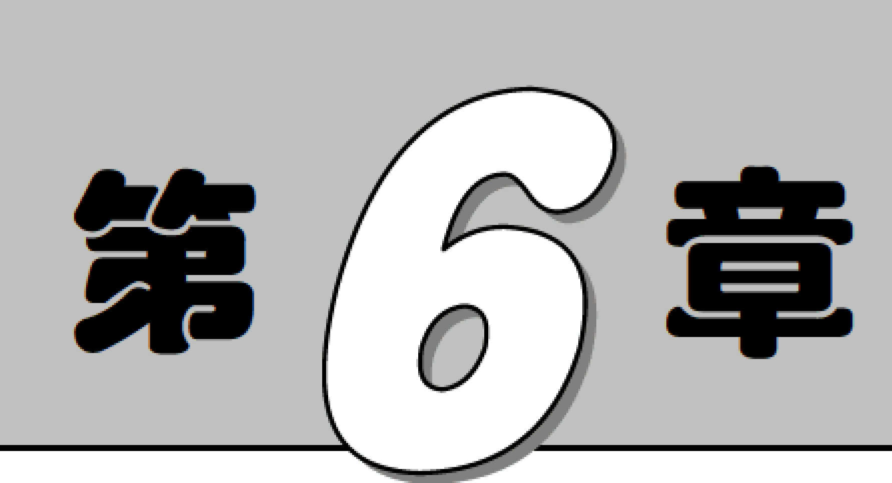
phpMyAdmin 图形化管理工具操作数据的关键点如下：

（1）查询数据信息：进入表名为 tb_demo（以 tb_demo 为例）的数据表界面，单击导航栏的“浏览”按钮。

（2）插入数据信息：单击导航栏“插入”按钮，根据提示插入信息。

（3）删除数据信息：单击删除字段的✕图标，删除一行数据。

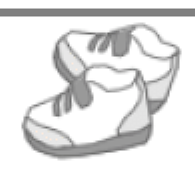
（4）修改数据信息：单击修改字段的✎图标，修改一个或多个字段数据。

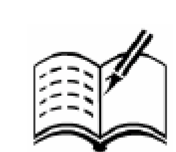


PHP 数据库编程

本章读者可以学到如下实例：

- 实例 098　通过 MySQL 函数访问数据库
- 实例 099　将数据以二进制形式上传到数据库
- 实例 100　查询日期型数据
- 实例 101　查询字符串
- 实例 102　使用 MySQL 存储过程实现用户注册
- 实例 103　使用 MySQL 事务处理实现银行安全转账
- 实例 104　避免输出中文字符串时出现乱码
- 实例 105　查询指定时间段的数据
- 实例 106　查询从指定位置的前 N 条记录
- 实例 107　通过 PHP 面向过程实现数据分页
- 实例 108　通过 PHP 面向对象实现数据分页
- 实例 109　查询结果不显示重复记录
- 实例 110　Delete 语句删除图书信息
- 实例 111　对统计结果进行排序
- 实例 112　使用 select 子句进行多表查询
- 实例 113　合并多个结果集
- 实例 114　简单的嵌套查询
- 实例 115　用 in 查询表中的记录信息
- 实例 116　使用聚集函数 sum()对学生成绩进行汇总
- 实例 117　使用聚集函数 avg 求学生的平均成绩
- 实例 118　使用聚集函数 min()求利润最少的商品
- 实例 119　使用聚集函数 max()求销售利润最高的商品
- 实例 120　使用聚集函数 count()求利润大于某值的数据
- 实例 121　多表联合查询
- 实例 122　left outer join 查询
- 实例 123　right outer join 查询
- 实例 124　利用 transform 分析数据
- 实例 125　使用格式化函数转换查询条件的数据类型
- 实例 126　在查询中使用日期函数
- 实例 127　一般搜索
- 实例 128　高级搜索
- 实例 129　程序员搜索引擎





实例 098　通过 MySQL 函数访问数据库

（实例位置：配套资源\SL\06\098）

实例说明

本实例讲解通过 PHP 中提供的 MySQL 函数来访问 MySQL 数据库。这里以开发一个电子商务网站为背景，通过 MySQL 函数实现与 MySQL 数据库的连接，然后读取 MySQL 数据库中的图书商品信息，并且将图书产品信息输出到页面中。其运行结果如图 6.1 所示。



图 6.1　通过 MySQL 函数访问数据库

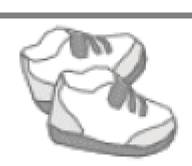
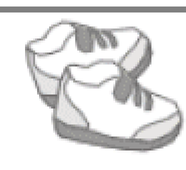
实现过程

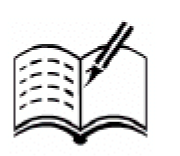
具体步骤如下：

（1）首先创建连接数据库的 conn.php 文件。然后将该文件存储在根目录下的 conn 文件夹中。将连接数据库文件单独提出的目的是为了日后维护方便，同时还可避免在其他需要连接数据库的页面中编写重复的代码，这样如果某个页面需要连接数据库，只需应用 include、require 或 include_once 等语句调用 conn.php 文件即可。

连接数据库首先应用的是 mysql_connect()函数获取与 MySQL 服务器的连接，返回一个标识符，然后应用 mysql_select_db()函数连接指定数据库（db_database06），最后应用 mysql_query()函数定义输出数据的编码格式为 GB2312，这样可以避免在输出中文字符串时出现乱码的问题。conn.php 文件的代码如下：

```
<?php
$conn=mysql_connect('localhost','root','111') or die('连接失败！ '.mysql_error());
if(mysql_select_db('db_database06',$conn))
    echo "";
    else
    echo ('数据库选择失败:' . mysql_error());
mysql_query("set names gb2312");
?>
```



（2）创建 index.php 文件，输出电子商务网站中的图书产品信息。首先应用 include_once 语句调用 conn.php 文件，实现与数据库的连接，然后应用 mysql_query()函数执行查询语句，查询出数据库中的图书产品信息，接着应用 mysql_fetch_array()函数将结果集以数组的形式返回，最后应用 while 循环语句，循环输出结果集中的数据。其关键代码如下：

```
<?php
    $result = mysql_query("select * from tb_mrbook",$conn);    //执行 SQL 指令
    while($myrow = mysql_fetch_array($result)){                //配合 while 输出数据库中的数据
?>
    <tr>
      <td align="left" bgcolor="#FFFFFF"> <?php echo $myrow[bookname];?></td>
      <td align="center" bgcolor="#FFFFFF"><?php echo $myrow[price];?></td>
      <td align="left" bgcolor="#FFFFFF"> <?php echo $myrow[synopsis];?></td>
      <td align="center" bgcolor="#FFFFFF"><?php echo $myrow[Maker];?></td>
    </tr>
<?php
  }
?>
```

技术要点

本实例主要应用 PHP 中提供的 MySQL 数据库函数，分别是 mysql_connect()、mysql_select_db()、mysql_query()、mysql_fetch_array()函数。

（1）mysql_connect()函数，打开一个到 MySQL 服务器的连接。如果成功则返回一个 MySQL 连接标识；失败则返回 false。

```
resource mysql_connect ( [string server [, string username [, string password [, bool new_link [, int client_flags]]]]] )
```

mysql_connect()函数的参数说明如表 6.1 所示。

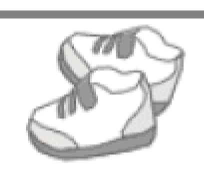
表 6.1　mysql_connect()函数的参数说明

参　　数	说　　明
server	MySQL 服务器。可以包括端口号，如"hostname:port"，或者到本地套接字的路径，如对于 localhost 的":/path/to/socket"。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是'localhost:3306'
username	用户名。默认值是服务器进程所有者的用户名
password	密码。默认值是空密码
new_link	如果用同样的参数第二次调用 mysql_connect()函数，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect()函数总是打开新的连接，甚至当 mysql_connect()函数曾在前面被用同样的参数调用过
client_flags	client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL，MYSQL_CLIENT_COMPRESS，MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE

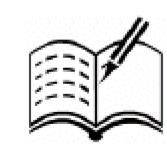
（2）mysql_select_db()函数，选择 MySQL 数据库。如果成功返回 true，失败返回 false。

```
bool mysql_select_db ( string database_name [, resource link_identifier] )
```





mysql_select_db()函数设定与指定的连接标识符所关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开的连接，本函数将无参数调用 mysql_connect()函数来尝试打开一个并使用。其后的每个 mysql_query()函数调用都会作用于活动数据库。



（3）mysql_query()函数，向与指定的连接标识符关联的服务器中的当前活动数据库发送一条 MySQL 查询。

```
resource mysql_query ( string query [, resource link_identifier] )
```

参数 query 为字符串类型，指定传入的 SQL 指令；参数 link_identfier 为资源类型，传入的是由 mysql_connect()函数或 mysql_pconnect()函数返回的连接号。如果省略该参数，则会使用最后一个打开的 MySQL 数据库连接。

脚下留神：

查询字符串不应以分号结束。

（4）mysql_fetch_array()函数，返回根据从结果集获取的行生成的数组，如果没有更多行则返回 false。

```
array mysql_fetch_array ( resource result [, int result_type] )
```

mysql_fetch_array()函数的参数说明如表 6.2 所示。

表 6.2　mysql_fetch_array()函数的参数说明

参　数	说　明
result	资源类型的参数，要传入的是由 mysql_query()函数返回的“数据指针”
result_type	可选参数，整数型参数，要传入的是 MYSQL_ASSOC、MYSQL_NUM、MYSQL_BOTH 3 种由 PHP 定义好的常数之一，默认值是 MYSQL_BOTH 用 MYSQL_ASSOC 只得到关联索引（相当于 mysql_fetch_assoc()函数） 用 MYSQL_NUM 只得到数字索引（相当于 mysql_fetch_row()函数） 用 MYSQL_BOTH 将得到一个同时包含关联和数字索引的数组

多学两招：

本函数返回的字段名是区分大小写的。

实例 099　将数据以二进制形式上传到数据库

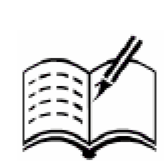
（实例位置：配套资源\SL\06\099）

实例说明

在进行网站开发的过程中，有一个问题是必须要考虑到的，即网站是否支持文件上传的功能？如果支持应该将文件存储在什么地方？是存储在服务器中，还是存储在数据库中？

具体将数据存储在什么地方，应该根据实际情况具体问题具体分析，如考虑文件的大





Note

小、类型等因素。在电子商务网站中，每种商品在展示的过程中都应该有效果图，而这个效果图的大小基本不会超过 5MB，类型为 bmp、jpg 或者 gif。对于这样的文件，我们就可以将其存储到数据库中。

本实例将以电子商务网站的商品效果图的上传为例，讲解如何将图片以二进制的形式存储到 MySQL 数据库中，并且查看数据库中存储的二进制格式的图片，其运行结果如图 6.2 所示。

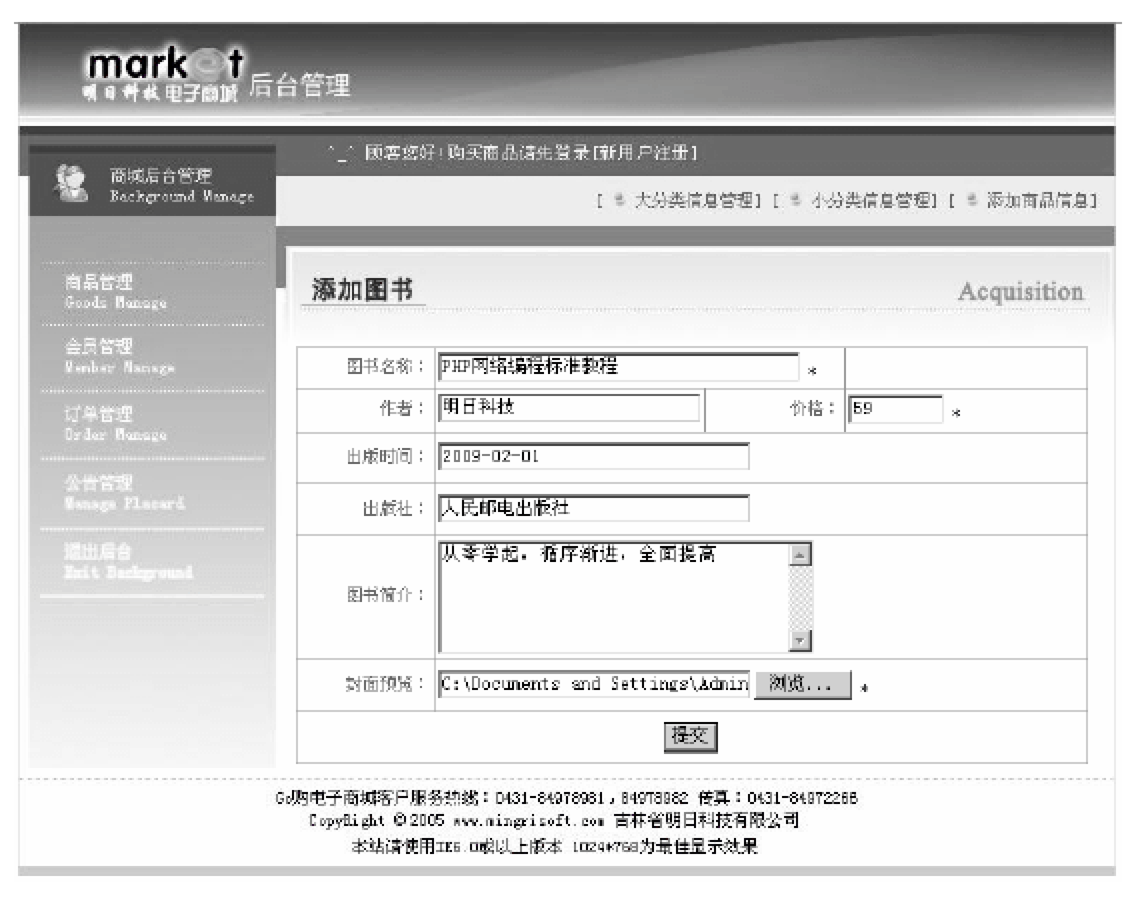




图 6.2　将数据以二进制形式上传到数据库中

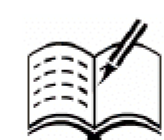
实现过程

具体步骤如下：

（1）创建 index.php 文件，添加表单，设置表单元素，通过 post 方法将图书产品的信息提交到 index_ok.php 文件中，同时编写 JavaScript 脚本对表单中提交的元素进行判断。其关键代码如下：

```
<script language="javascript">
function check_form(form){
  if(form.bookname.value==""){
        alert("图书名称不能为空!!");
        form.bookname.select();
          return(false);
  }
  if(form.price.value==""){
        alert("图书价格不能为空!!");
        form.price.select();
          return(false);
  }
  if(form.cover.value==""){
        alert("图书封面不能为空!!");
        form.cover.select();
          return(false);
  }
   return(true);
}
```



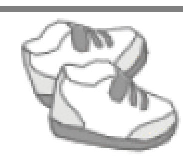
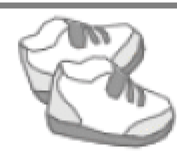


```
</script>
<form name="form" method="post" action="index_ok.php" onsubmit="return check_form(this)">
  <tr>
    <td height="36" align="right" bgcolor="#FFFFFF">图书简介：</td>
    <td colspan="3" bgcolor="#FFFFFF">
<textarea name="synopsis" cols="35" rows="5" id="synopsis"></textarea></td>
  </tr>
  <tr>
    <td height="36" align="right" bgcolor="#FFFFFF">封面预览：</td>
    <td colspan="3" bgcolor="#FFFFFF"><input name="cover" type="file" id="cover" size="30">
      <span class="STYLE1">*</span></td>
  </tr>
  <tr>
    <td height="36" colspan="4" align="center" bgcolor="#FFFFFF">
<input type="submit" name="Submit" value="提交"></td>
  </tr>
</form>
```

（2）创建连接数据库的 conn.php 文件。

（3）创建 index_ok.php 文件。首先通过 include 语句调用连接数据库的 conn.php 文件，然后通过$_POST[]方法获取表单中提交的数据，最后编写 insert 语句将表单中提交的数据添加到指定的数据表中。其关键代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");                //设置编码格式
include("conn/conn.php");                                        //包含数据库连接文件
if(isset($_POST["Submit"])){
  $bookname=$_POST["bookname"];
  $price=$_POST["price"];
  $maker=$_POST["maker"];
  $issuDate=$_POST["issuDate"];
  $publisher=$_POST["publisher"];
  $synopsis=$_POST["synopsis"];
  $cover=$_FILES["cover"]["name"];                         //获取上传文件名称
    $cover_type=strstr($cover,".");                        //获取从"."到最后的字符
  if($cover_type != ".jpg" && $cover_type != ".gif" && $cover_type != ".JPG" && $cover_type !=
".GIF" && $cover_type != ".bmp" && $cover_type != ".BMP"){   //判断图片的格式
      echo "<script>alert('封面图片格式不正确，请进行处理后再上传！'); window.location.href=
'index.php';</script>";
  }else{
      $fp=fopen($_FILES['cover']['tmp_name'],"rb");        //以二进制格式打开文件
        $image=addslashes(fread($fp,filesize($_FILES['cover']['tmp_name'])));  //读取文件
      $sql="insert into tb_book(bookname,price,maker,issuDate,publisher,synopsis,cover)values
('$bookname','$price','$maker','$issuDate','$publisher','$synopsis','$image')";      //执行添加操作
      $result=mysql_query($sql,$conn);
      if($result==true){
          echo "文件上传成功!!";
          echo "<meta http-equiv=\"refresh\" content=\"30 url=index.php\">";
      }else{
          echo "上传失败!!";
```

```
                    echo "<meta http-equiv=\"refresh\" content=\"30 url=index.php\">";
                }
            }
        }
?>
```



技术要点

本实例主要应用表单中的文件域提交图片文件。表单中的文件域代码格式如下：

```
<input name="cover" type="file" id="cover" size="30">
```

type 指定表单元素的类型为文件域；name 指定表单元素的名称；size 指定表单元素的大小。

通过$_POST[]方法获取表单中提交的数据，并应用 insert 语句将表单中的数据添加到数据表中。Insert 语句的语法格式如下：

```
Insert into table_name (column_name,column_name2, … ) values (value1, value2, … )
```

参数说明：

☑ table_name：数据表的名称。

☑ column_name：数据表中字段的名称。

☑ valuel：表单提交的变量。

脚下留神：

在 Insert 语句中，column_name 与 value1 一定要相互对应；column_name 与数据表中的字段也要相互对应。

实例 100　查询日期型数据

（实例位置：配套资源\SL\06\100）

实例说明

对日期型数据进行查询在商业网站中得到了广泛的应用，例如，查询某范围员工的出生日期、商品的进货时间等。本实例将在图书信息表中查询图书出版日期为“2010-07-19”的图书信息，运行本实例，如图 6.3 所示。

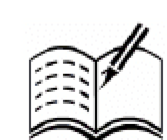
查询日期型数据

查询数据

ID	书名	剩余	日期
1	《PHP范例大全》	101	2010-07-19
2	《JAVA范例大全》	150	2010-07-19
3	《VB范例大全》	180	2010-07-19
4	《C++范例大全》	178	2010-07-19

图 6.3　查询日期型数据





实现过程

创建 index.php 文件，连接 MySQL 数据库，当单击显示数据按钮时，程序会自动显示出版日期在“2010-07-19”这一天的所有图书。其代码如下：

```
<?php
  if($_POST[sub]){
//单击按钮
        $conn = mysql_connect("localhost","root","111") or die("connect mysql false");//连接 MySQL
        mysql_select_db("db_database06",$conn) or die("connect database false"); //连接数据库
        mysql_query("set names utf8");                                  //设置编码格式
        $rs = mysql_query("select * from tb_demo036 where date = '2010-07-19'");//返回结果集
?>
        <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG"> 书 名 </td><td background="pic/head.JPG"> 剩 余 </td><td background="pic/head.JPG">日期</td></tr>
<?php
        while($rst = mysql_fetch_row($rs)){                             //循环输出结果
            echo "<tr><td background='pic/head.JPG'>".$rst[0]."</td><td background='pic/head.JPG'>".$rst[1]."</td><td background='pic/head.JPG'>".$rst[2]."</td><td background='pic/head.JPG'>".$rst[3]."</td></tr>";
        }
  }
?>
```

技术要点

不同的数据库对日期型数据的查询是有区别的，下面将以几种典型的数据库为例讲解在不同的数据库中对日期型数据的查询方式。

下面以在学生表（tb_student）中查询出生日期（birthday）为“1998-01-01”为例进行讲解。

（1）MySQL 数据库中对日期型数据的查询。

```
select * from tb_student where birthday='1998-01-01'
```

（2）SQL Server 数据库中对日期型数据进行查询。

```
select * from tb_student where birthday='1998-01-01'
```

（3）Access 数据库中对日期型数据进行查询。

```
select * from tb_student where birthday=#1998-01-01#
```

通过上面 3 个例子可以发现，在 MySQL 数据库和 SQL Server 数据库中实现对日期型数据查询所要查询的日期应用单引号括起来，而在 Access 数据库中使用 JET SQL 语法查询时所查询的日期应用“#”号括起来。

本实例实现查询日期型数据的 SQL 语句代码如下：

```
select * from tb_demo036 where date = '2010-07-19';
```

查询 tb_demo036 数据表中所有时间 date 等于“2010-07-19”的数据。





Note

实例 101　查询字符串

（实例位置：配套资源\SL\06\101）

实例说明

对字符串进行查询是项目开发过程应用几率最高的查询，并且这种查询经常与通配符配合使用实现信息的匹配查询。运行本实例，如图 6.4 所示。



图 6.4　查询字符串

实现过程

创建 index.php 文件，连接 MySQL 数据库，显示查询到的图书信息。其代码如下：

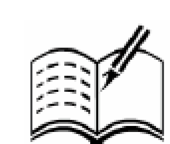
```
<?php
  if($_POST[sub]){                                                          //单击按钮
        $conn = mysql_connect("localhost","root","111") or die("connect mysql false");//连接 MySQL
        mysql_select_db("db_database06",$conn) or die("connect database false"); //连接数据库
        mysql_query("set names utf8");                                      //设置编码格式
        //返回结果集
        $rs = mysql_query("select * from tb_demo036 where name like '%$_POST[select]%'");
?>
        <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">书名</td><td background="pic/head.JPG">剩余</td><td background="pic/head.JPG">日期</td></tr>
<?php
        while($rst = mysql_fetch_row($rs)){                                 //循环输出
              echo "<tr><td background='pic/head.JPG'>".$rst[0]."</td><td background='pic/head.JPG'>".$rst[1]."</td><td background='pic/head.JPG'>".$rst[2]."</td><td background='pic/head.JPG'>".$rst[3]."</td></tr>";
        }
  }
?>
```

技术要点

SQL 语句可以对字符串进行完全匹配查找和模糊查找，如果对字符串进行匹配查找则直接用等号作为查询条件的连接谓词；如果对字符串进行模糊查找则用 like 关键字作为连接谓词。下面将通过具体实例讲解如何实现对字符串的查找。

（1）从学生成绩表（tb_score）中查询名为“小刘”的学生信息，其中学生姓名字段





为 sname。

```
select * from tb_score where sname='小刘'
```

（2）从学生表（tb_student）中查询所有姓“刘”的学生信息，其中学生姓名字段为 sname。

```
select * from tb_student where sname like '刘%'
```

（3）从职工表（tb_worker）中查询所有职务为“程序员”的员工信息，其中员工职务字段为 zhiwu。

```
select * from tb_worker where zhiwu like '%程序员'
```

（4）从图书信息表（tb_book）中查询所有 PHP 类相关图书，其中书名字段为 bookname。

```
select * from tb_book where bookname like '%PHP%'
```

实例 102　使用 MySQL 存储过程实现用户注册

（实例位置：配套资源\SL\06\102）

实例说明

存储过程是 MySQL5.0 以后的版本中才支持的技术，通过存储过程的应用可以提高系统的运行效率。本实例将向读者介绍 MySQL5.0 版本中存储过程的创建以及 PHP 调用 MySQL 存储过程的方法。首先在命令提示符下创建存储过程，然后运行本实例，如图 6.5 所示，在图中的文本框中输入注册信息后，单击“注册”按钮即可将用户的注册信息保存到数据库中。



图 6.5　填写用户注册信息

实现过程

具体步骤如下：

（1）本实例应用存储过程实现用户登录，这里首先创建数据表，并添加字段信息。本实例中使用的是 tb_user 表。数据表中的字段属性说明如图 6.6 所示。

（2）接下来创建存储过程。首先打开“命令提示符”，进入到命令提示符后，在命令提示符中依次编写如图 6.7 所示的内容。




Note


服务器: localhost ▸ 数据库: db_database06 ▸ 表 : tb_user

浏览 结构 SQL 搜索 插入 导出 Import 操作

	字段	类型	整理	属性	Null	
☐	id	int(10)			否	关键字
☐	username	varchar(50)	gb2312_chinese_ci		否	用户名
☐	password	varchar(50)	gb2312_chinese_ci		否	密码
☐	email	varchar(50)	gb2312_chinese_ci		否	邮箱
☐	address	varchar(80)	gb2312_chinese_ci		否	地址

图 6.6　tb_user 表的字段属性说明

```
C:\WINDOWS\system32\cmd.exe - mysql -uroot -proot
C:\Documents and Settings\Administrator>mysql -uroot -proot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 27
Server version: 5.0.67-community-nt-log MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use db_database06
Database changed
mysql> delimiter //
mysql> create procedure pro_regs(in nc varchar(50),in pwd varchar(50),in email v
archar(50),in address varchar(50))
    -> begin
    -> insert into tb_user(username,password,email,address)values(nv,pwd,email,a
ddress);
    -> end;
    -> //
Query OK, 0 rows affected (0.03 sec)
```

图 6.7　创建存储过程

创建存储过程的关键代码如下：

```
/*  使用的存储过程
delimiter  //
create procedure pro_regs(in nc varchar(50),in pwd varchar(50),in email varchar(50),in address varchar(50))
begin
insert into tb_user(username,password,email,address)values(nc,pwd,email,address);
end;
//
*/
```

（3）创建 index.php 文件，添加表单，设置表单元素，通过 post 方法将用户注册信息提交到本页，代码如下：

```
<form name="form1" method="post" action="index.php"   onsubmit="return chkinput(this)">
    <tr>
      <td  width="150"  height="25"  bgcolor="#FFFFFF"><div  align="center">用户昵称：</div></td>
    <td width="347" bgcolor="#FFFFFF"> <input type="text" name="nc" class="inputcss" size="25"></td>
    </tr>
    <tr>
      <td height="25" bgcolor="#FFFFFF"><div align="center">注册密码：</div></td>
      <td   height="25"   bgcolor="#FFFFFF"> <input   type="password"   name="pwd"
```



```
class="inputcss" size="25"></td>
                    </tr>
                    <tr>
                      <td    height="25"    colspan="2"    bgcolor="#FFFFFF"><div    align="center"><input
type="submit" name="submit" value="注册" class="buttoncss">  <input type="reset" value="重写
" class="buttoncss"></div></td>
                    </tr>
               </form>
```



通过 PHP 预定义类 mysqli 实现与 MySQL 数据库的连接，并通过 call 语句调用存储过程 pro_regs，将用户注册的数据添加到 tb_user 数据表中。其关键代码如下：

```
<?php
 if($_POST[submit]!="")
 {
     $conn=new mysqli("localhost","root","111","db_database06");     //连接数据库
     $conn->query("set names gb2312");                                //设置编码格式
     $nc=$_POST[nc];                                                  //获取用户注册提交的数据
     $pwd=md5($_POST[pwd]);
     $email=$_POST[email];
     $address=$_POST[address];
     if($sql=$conn->query("call  pro_reg('".$nc."','".$pwd."','".$email."','".$address."')")){// 调用并执
行存储过程
      echo "<script>alert('用户注册成功!');</script>";
  }else{
        echo "<script>alert('用户注册失败!');</script>";
  }

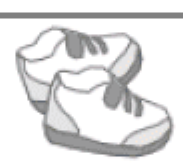
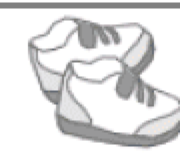
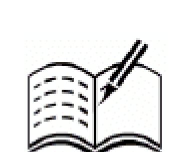
 }
?>
```

技术要点

本实例主要应用存储过程实现用户注册的功能，其关键在于存储过程的创建。MySQL 存储过程的语法如下：

```
create procedure proc_name (in parameter integer)
begin
declare variable varchar(20);
if parameter=1 then
set variable='MySQL';
else
set variable='PHP';
end if;
insert into tb (name) values (variable);
end;
```

MySQL 中存储过程的建立以关键字 create procedure 开始，后面紧跟存储过程的名称和参数。MySQL 的存储过程名称不区分大小写，例如，PROCE1()和 proce1()代表同一存储过程名。存储过程名不能与 MySQL 数据库中的内建函数重名。

存储过程的参数一般由三部分组成。第一部分可以是 in、out 或 inout。in 表示向存储过程中传入参数；out 表示向外传出参数；inout 表示定义的参数可传入存储过程并可以被存储过程修改后传出存储过程，存储过程默认为传入参数，所以参数 in 可以省略。第二部分为参数名。第三部分为参数的类型，该类型为 MySQL 数据库中所有可用的字段类型，如果有多个参数，参数之间可以用逗号进行分割。

MySQL 存储过程的语句块以 begin 开始，以 end 结束。语句体中可以包含变量的声明、控制语句、SQL 查询语句等。由于存储过程内部语句要以分号结束，所以在定义存储过程前应将语句结束标志“;”更改为其他字符，并且该字符应该在存储过程中出现的几率也应该较低，可以用关键字 delimiter 更改，例如：

```
mysql>delimiter //
```

存储过程创建之后，可用如下语句进行删除，参数 proc_name 指存储过程名。

```
drop procedure proc_name
```

创建及调用存储过程的步骤如下：

（1）MySQL 存储过程是在命令提示符下创建的，所以首先应该打开“命令提示符”。

（2）进入“命令提示符”后，首先应该登录 MySQL 数据库服务器，在命令提示符下输入如下命令：

```
mysql –u 用户名  –p 用户密码
```

（3）更改语句结束符号，本实例将语句结束符更改为“//”，代码如下：

```
delimiter //
```

（4）创建存储过程前应首先选择某个数据库，代码如下：

```
use  数据库名
```

（5）创建存储过程。

（6）通过 call 语句调用存储过程。

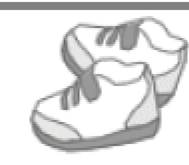
实例 103　使用 MySQL 事务处理实现银行安全转账

（实例位置：配套资源\SL\06\103）

实例说明

事务处理机制在程序开发过程中有着非常重要的作用，它可以使整个系统更加安全，例如，在银行处理转账业务时，如果 A 账户中的金额刚被发出，而 B 账户还没来得及接受就发生停电，这会给银行和个人带来很大的经济损失。采用事务处理机制，一旦在转账过程中发生意外，则程序将回滚，不做任何处理。

本实例设计了一个模拟的银行转账系统，并且应用事务处理机制保证转账操作安全、顺利地进行。运行本实例，如图 6.8 所示，在图中的文本框中输入要转给 B 账户的金额后，单击“转账”按钮即可实现转账。





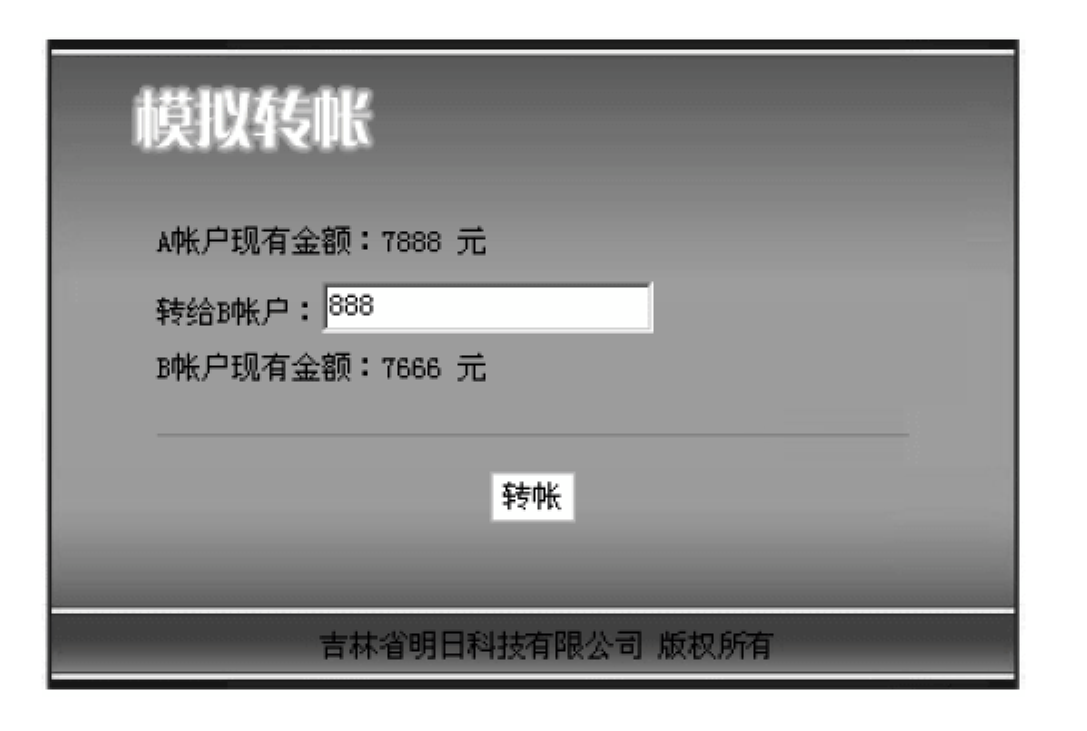


图 6.8　应用事务回滚模拟银行转账

实现过程

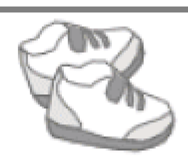
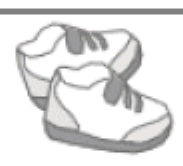
具体步骤如下：

（1）创建数据库连接文件 conn.php。这里应用 PHP 中的预定义类 mysqli 连接数据库，代码如下：

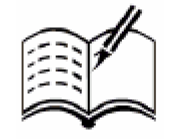
```
<?php
$conn=new mysqli("localhost","root","111","db_database06");
$conn->query("set names gb2312");
?>
```

（2）创建 index.php 文件，读取数据库中的数据。首先通过 include 语句调用连接数据库文件。然后执行查询语句，读取并输出数据库中的数据。最后创建表单，添加文本框提交转账金额，将数据提交到 index_ok.php 页面中。其关键代码如下：

```
<?php
$conn=new mysqli("localhost","root","111","db_database06");
$conn->query("set names gb2312");
?>
 <form  name="form1"  method="post"  action="index_ok.php"  onsubmit="return  chkinput(this)">
<table width="300" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
        <tr>
          <td>A 账户现有金额：<?php echo $info[money];?> 元</td>
        </tr>
      </table>
        <table width="300" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
          <tr>
            <td>转给 B 账户：<input type="text" name="tob" class="inputcss" size="20"></td>
          </tr>
        </table>
        <table width="300" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
          <tr>
            <td>B 账户现有金额：<?php echo $infos[money];?> 元</td>
          </tr>
        </table>
        <table width="300" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
          <tr>
```

```
            <td><div align="center"><input type="submit" value="转账" class="buttoncss"></div></td>
          </tr>
        </table>
    </form>
```



Note

（3）创建 index_ok.php 文件，获取表单中提交的数据，应用事务处理机制完成转账操作，代码如下：

```
<?php
$tob=$_POST[tob];
include_once("conn/conn.php");
$conn->autocommit(false);                    //关闭自动提交
if(!$conn->query("update tb_bank set money=money-'".$tob."' where username='mrsoft'"))
{
 $conn->rollback();                          //实现事务回滚
}
if(!$conn->query("update tb_bank set money=money+'".$tob."' where username='mr'"))
{
 $conn->rollback();
}
$conn->commit();                             //提交所有更新语句
$conn->autocommit(true);
echo "<script>window.location.href='index.php';</script>";
?>
```

技术要点

事务的作用在服务器发生错误或崩溃的情况下确保数据库的一致性。事务是一个或一系列的查询，这些查询要么全部执行，要么全部不执行。例如，银行转账需要两个过程来完成，首先需从某个账户扣除一定金额，之后在另一个账户中增加相同的金额。如果这两个过程不同时执行，从第一个账户中扣除金额后还没来得及增加第二个账户的金额就发生停电或服务器崩溃，这将给用户和银行造成很大的损失。如果采用事务处理上述过程，即使出现上述事故也不会给用户造成损失。

一个事务被永久的写入到数据库中称事务提交，将状态重置到事务开始之前的状态称为事务回滚。

事务具有以下 4 个特性：

（1）原子性：指事务作为一个整体要么完全执行，要么完全不执行。

（2）一致性：指一个事务必须能够使数据处于一致的状态。

（3）孤立性：在事务完全完成之前，它们都是孤立的。

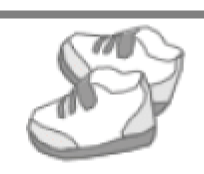
（4）持续性：一旦写入数据库后，事务必须是永久和持续的。

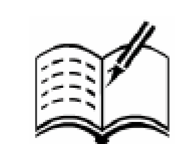
PHP 处理 MySQL 事务是通过 PHP 中的预定义类 mysqli 来完成的，具体方法如下：

（1）autocommit()：该方法的参数为 false 或 true，false 表示禁止自动提交查询，true 表示自动提交查询。

（2）rollback()：该方法的作用是当发生意外时，自动回滚到事务开始之前的状态。

（3）commit()：该方法的作用是提交事务。





在本实例中应用事务实现转账的代码如下：

```
<?php
$tob=$_POST[tob];
include_once("conn.php");
$conn->autocommit(false);
if(!$conn->query("update tb_zy set money=money-'".$tob."' where flag='mrsoft'"))
{
  $conn->rollback();
}
if(!$conn->query("update tb_zy set money=money+'".$tob."' where flag='mr'"))
{
  $conn->rollback();
}
$conn->commit();
$conn->autocommit(true);
echo "<script>window.location.href='index.php';</script>";
?>
```

首先调用 mysqli 类的 autocommit()方法关闭数据库的自动提交，然后减少 A 账户一定数量的金额并判断查询是否顺利执行，如果在此过程中发生意外，则通过调用 mysqli 类的 rollback()方法回滚，不做任何处理，反之为 B 账户增加相同的金额。最后通过调用 mysqli 类的 commit()方法提交查询，实现转账。

实例 104　避免输出中文字符串时出现乱码

（实例位置：配套资源\SL\06\104）

实例说明

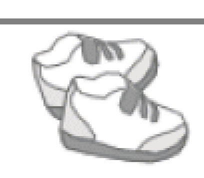
将数据库中信息显示到网页中经常会遇到这样的问题，网页中输出的中文文字为乱码。造成问题的主要原因是数据库中的编码格式与页面设置的编码格式不符或者没有将数据信息统一为 GBK、GB2312 或 UTF-8。本实例通过设置编码格式向用户演示如何避免输出中文字符串时出现乱码，运行结果如图 6.9 所示。

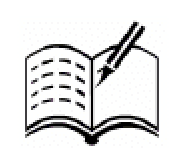
避免输出中文字符串时出现乱码

数据库编码格式为UTF-8　输入编码格式　确定

ID	书名	价格	日期
1	PHP????□▯ ???伯td>	80□	2010-07-17
2	VB????□▯ ???伯td>	80□	2010-07-17
3	JAVA????□▯ ???伯td>	79□	2010-07-17
4	JAVAWEB????□▯ ???伯td>	78□	2010-07-17
5	.NET????□▯ ???伯td>	78□	2010-07-17
6	C#????□▯ ???伯td>	78□	2010-07-17
7	C++????□▯ ???伯td>	78□	2010-07-17

图 6.9　错误设置编码格式输出中文





Note

实现过程

创建脚本文件，并命名为 index.php。创建 form 表单，将要设置的编码格式提交到本页。在本页中连接服务器，连接指定的数据库，根据 form 表单提交的数据设置数据库的编码格式，然后循环输出数据库中的数据，其代码如下：

```
<?php
 if($_POST[sub]){                                          //单击“确定”按钮
     if($_POST[name] == "" || $_POST[name] == "输入编码格式"){//如果文本框为空或初始值
         echo "<script>alert('文本框内容不正确');</script>";
     }else{
     $conn = mysql_connect("localhost","root","111") or dir ("connect MySQL false");//连接
MySQL 数据库
         mysql_select_db("db_database06",$conn) or die ("connect database false");//连接数据库
         mysql_query("set names $_POST[name]");              //设置编码格式
         $rs = mysql_query("select * from tb_demo031");      //返回结果集
         echo "<table width='580px'><tr><td bgcolor='#FF0000'>ID</td><td bgcolor='#FF0000'>
书名</td><td bgcolor='#FF0000'>价格</td><td bgcolor='#FF0000'>日期</td></tr>";
         while($rst = mysql_fetch_row($rs)){                 //循环输出
             echo "<tr><td bgcolor='#FF0000'>".$rst[0]."</td><td bgcolor='#FF0000'>".$rst[1].
"</td><td bgcolor='#FF0000'>".$rst[2]."</td><td bgcolor='#FF0000'>".$rst[3]."</td></tr>";
         }
         echo "</table>";
     }
 }
?>
```

技术要点

set 语句：用于设置不同变量。其语法如下：

```
SET variable_assignment [, variable_assignment] ...
variable_assignment:
      user_var_name = expr
    | [GLOBAL | SESSION] system_var_name = expr
    | @@[global. | session.]system_var_name = expr
```

本实例通过 set 语句设置页面的编码格式，其代码如下：

```
set names $_POST[name];
```

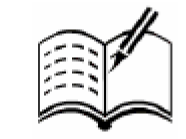
实例 105　查询指定时间段的数据

（实例位置：配套资源\SL\06\105）

实例说明

在图书管理系统中，经常需要查询指定出版日期范围内的图书情况。运行本实例，如图 6.10 所示，首先在图中的文本框中输入日期范围，然后单击“查看”按钮即可实现查找该日期范围内的所有图书信息。





查询指定时间段的数据

输入开始日期　输入结束日期　查看　日期格式：xxxx-xx-xx

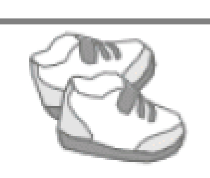
ID	书名	价格	日期
4	JAVAWEB开发实战宝典	78元	2010-01-01
3	JAVA开发实战宝典	79元	2010-03-11
2	VB开发实战宝典	80元	2010-04-11
6	C#开发实战宝典	78元	2010-04-22
1	PHP开发实战宝典	80元	2010-05-17

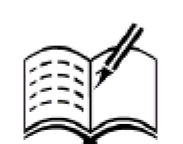
图 6.10　查询指定时间段的数据

实现过程

创建 index.php 文件，启动 MySQL 服务并连接 MySQL 数据库。当单击“查看”按钮时，显示指定时间段的数据。代码如下：

```
<?php
 if($_POST[sub]){                                                    //单击“查看”按钮
                                                                     //判断文本框信息
     if($_POST[text1] == "" || $_POST[text1] == "输入开始日期" || $_POST[text] == "" || $_POST[text] == "输入结束日期"){
         echo "<script>alert('请输入日期');</script>";               //JavaScript 提示
     }else{
                                                                     //验证日期格式的正则表达式
 if(preg_match("/([0-9]{3}[1-9][0-9]{2}[1-9][0-9]{1}[1-9][0-9]{2}|[1-9][0-9]{3})-(((0[13578]|1[0-2])-(0[1-9]|[12][0-9]|3[01]))|((0[469]|11)-(0[1-9]|[12][0-9]|30))|(02-(0[1-9]|[1][0-9]|2[0-8])))/",$_POST[text])){
     //连接 MySQL
             $conn = mysql_connect("localhost","root","111") or die("connect mysql false");
     //连接 MySQL 数据库
             mysql_select_db("db_database06",$conn) or die("connect database false");
             mysql_query("set names utf8");                          //设置编码格式
             $rss = mysql_query("select * from tb_demo047 where date between '$_POST[text1]' and '$_POST[text]' order by date");  //返回结果集
             echo'<table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">书名</td><td background="pic/head.JPG">价格</td><td background="pic/head.JPG">日期</td></tr>';
             while($rst = mysql_fetch_array($rss)){                  //循环输出结果
?>
     <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background="pic/head.JPG"><?php echo $rst[3];?></td></tr>
<?php
             }
         }else{
     //JavaScript 提示信息
             echo "<script>alert('日期格式错误！正确格式为：xxxx-xx-xx');</script>";
         }
     }
 }
?>
```





技术要点

在 SQL 语句中实现对指定范围内的数据进行查找可以通过以下两种方式实现：

第一可以通过关键字 between…and…实现，语法格式如下：

```
select 要查找的字段 from 表名 where 字段名 between 初始值 and 终止值
```

第二种方式可以通过比较运算符实现，语法格式如下：

```
select 要查找的字段 from 表名 where 字段名>初始值 and 字段名<终止值
```

实例 106　查询从指定位置的前 N 条记录

（实例位置：配套资源\SL\06\106）

实例说明

查询从指定位置开始显示 N 条记录，这也是 PHP 实现分页显示的原理。本实例将应用 MySQL 中的 limit 关键字，完成从指定位置开始显示 N 条记录的查询操作。从第 4 条记录开始，查询两条数据，其运行结果如图 6.11 所示。

查询从指定位置的前N条记录

4　2　查看

ID	书名	价格	日期
5	.NET开发实战宝典	78元	2010-07-17
6	C#开发实战宝典	78元	2010-07-17

图 6.11　查询从指定位置的 N 条记录

实现过程

创建 index.php 文件，编写网页框架，启动 MySQL 服务器，连接 MySQL 数据库，当单击“查看”按钮时，显示指定位置的 N 条记录。代码如下：

```
if($_POST[sub]){                                                    //单击按钮
    //判断文本框信息
    if($_POST[text] == "" || $_POST[text] == "输入图书名称" || $_POST[text1] == "" || $_POST[text1]
== "输入查询条数"){
            echo "<script>alert('输入图书名');</script>";             //JavaScript 提示
    }else{
            if(preg_match("/\d/",$_POST[text])){                    //正则表达式验证
                    if(preg_match("/\d/",$_POST[text1])){           //正则表达式验证
                            //连接 MySQL
                            $conn = mysql_connect("localhost","root","111") or die("connect mysql false");
                            //连接 MySQL 服务器
                            mysql_select_db("db_database06",$conn) or die("connect database false");
                            mysql_query("set names utf8");          //设置编码格式
                            //返回结果集
```





```
                    $rss = mysql_query("select * from tb_demo031   limit $_POST[text1], $_POST
[text]");
                    echo'<table   width="580px"><tr><td     background="pic/head.JPG">ID</td><td
background="pic/head.JPG">书名</td><td   background="pic/head.JPG">价格</td><td   background="pic/
head.JPG">日期</td></tr>';
                    while($rst = mysql_fetch_array($rss)){                    //循环输出结果
       ?>
            <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.
JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background=
"pic/head.JPG"><?php echo $rst[3];?></td></tr>
       <?php
                    }
                }else{
                    echo "<script>alert('输入的不是一个数字');</script>";   //JavaScript 提示
                }
            }else{
                echo "<script>alert('输入的不是一个数字');</script>";       //JavaScript 提示
            }
        }
      }
      ?>
```

指点迷津：

本实例在运行时，用户设定从第 2 条开始输出，而显示数据时是从第 3 条开始，出现这种情况的原因是数据信息 ID 是从 0 开始计算的。

技术要点

实现从指定位置开始查询满足条件的 n 条记录，主要应用 MySQL 的扩展关键字 limit，该关键字的使用格式如下：

```
select 要查询的字段 from 表名 where 查询的条件 limit 满足条件的起始位置，记录的个数
```

关键字 limit 后有两个参数：第一个参数用于指定要满足条件记录的起始位置；第二个参数指定查询结果中满足条件的记录个数。

本实例实现查询指定位置的 N 条记录的 SQL 语句代码如下：

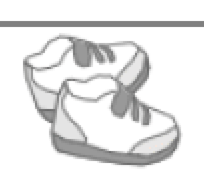
```
select * from tb_demo031   limit $_POST[text1], $_POST[text];
```

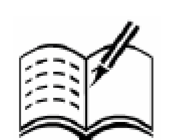
实例 107　通过 PHP 面向过程实现数据分页

（实例位置：配套资源\SL\06\107　视频位置：配套资源\SP\06\107）

实例说明

对于数据库中存储的大量的数据，如果想要进行输出，最佳的方法就是使用分页。通过分页输出数据，可以保持页面整洁，同时提高数据的浏览速度。





本实例中将介绍如何通过 PHP 面向过程编程实现数据的分页输出。运行结果如图 6.12 所示。



图 6.12　分页输出数据库中数据

在本实例中综合了上下分页和跳转分页的功能。上下分页以当前页码为基础，实现上一页和下一页的跳转，并且输出上一页的最后一条数据和下一页的第一条数据的内容；而跳转分页实现以十页为一个单位，实现上 10 页和下 10 页的跳转，并且可以在当前页显示的 10 个超链接中任意跳转。

实现过程

具体步骤如下：

（1）创建数据库连接文件 conn.php。通过 PHP 中的 MySQL 函数连接 MySQL 数据库，代码如下：

```
<?php
$id=mysql_connect("localhost","root","111")or dir('连接失败:' . mysql_error());
if(mysql_select_db("db_database06",$id))
echo "";
else
echo ('连接失败:' . mysql_error());
mysql_query("set names gb2312");
?>
```

（2）创建 index.php 文件，实现分页输出数据库中的数据。

首先，调用连接数据库文件，初始化分页变量。代码如下：

```
<?php
   include_once("conn/conn.php");
if ($_GET[page]=="")
{
$_GET[page]=1;
};
```



```
if ($_GET[link_type]=="")
{
$_GET[link_type]=0;};
?>
```

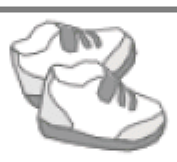
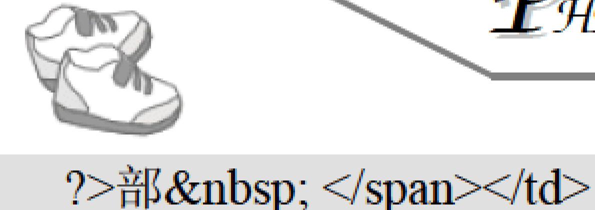
然后，读取数据库中的数据，应用分栏方法输出数据库中的数据，并且定义数据分页输出的变量。其关键代码如下：

```
<?php
 $query_1 = mysql_query ( "select * from tb_book" );              //获取数据库中总的记录数
 if (mysql_num_rows ( $query_1 ) <= 0) {                          //判断值是否为空
        ?>
<tr><td colspan="2" align="center">没有数据！ </td></tr>
<?php
 } else {
        $number = 0;                                                 //定义变量
        if ($_GET [page]) {
               $page_size = 2;                                       //每页显示两条数据
               $query = "select count(*) as total from tb_book";
               $result = mysql_query ( $query );
               $message_count = mysql_result ( $result, 0, "total" ); //统计总的记录数
               $page_count = ceil ( $message_count / $page_size );   //计算总共有几页
               $offset = ($_GET [page] - 1) * $page_size;            //获取当前页的起始页
               $query_2 = mysql_query ( "select * from tb_book limit $offset, $page_size" );
               while ( $myrow_2 = mysql_fetch_array ( $query_2 ) ) {  //循环输出
                      if (($number % 2) == 0) {                      //取模
                             ?>
    <tr>
<?php
                      }
                      ?>
                   <td colspan="2" align="center"><!--输出数据库中数据，省略了部分代码--></td>
<?php
                      if (($number % 2) != 0) {                      //取模
                             ?>
</tr>
<?php
                      }
                      $number ++;                                    //增加变量值
               }
        }

 }
 ?>
```

接着，以当前页码为基础，实现上一页和下一页的跳转，输出上一页的最后一条数据和下一页的第一条数据。其关键代码如下：

```
<td width="70" height="22" align="center" bgcolor="#FFFFFF"><spanclass="STYLE2">图书：
<?php
 echo $message_count;
```

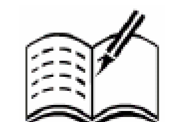



```
?>部  </span></td>
<td width="139" align="center" bgcolor="#FFFFFF">上一部：<a
  href="index.php?page=<?php
  if ($_GET [page] == "" or $_GET [page] <= 1) {          //判断如果分页变量值为空或小于 1
        echo 1;                                           //输出 1
  } else {
        echo ceil ( ($_GET [page] * 2 - 2) / 2 );//否则，根据每页中显示两条记录，计算出上一页 page
的值
  }
  ?>">
<?php
  $page_3 = ($_GET [page] - 2) * 2 + 1;                   //计算出上一页中两条记录的结束值
  if ($page_3 <= 0) {                                     //如果$page_3 的值小于等于 0，则输出 0
        $page_3 = 0;
  }
  $query_3 = mysql_query ( "select * from tb_book limit $page_3,1" );
  $myrow_3 = mysql_fetch_array ( $query_3 );
  echo $myrow_3 [bookname];                               //输出上一页中的最后一条数据
  ?>
          </a></td>
<td width="126" align="center" bgcolor="#FFFFFF">下一部：<a
  href="index.php?page=<?php
  if ($_GET [page] == "" or $_GET [page] <= 1) {//判断如果 page 的值等于空或者小于等于 1 则输出 2
        echo 2;
  } else if ($_GET [page] >= $page_count) {
        echo $page_count;                        //判断如果 page 的值超过总记录则输出总记录数
  } else {
        echo ceil ( ($_GET [page] * 2 + 1) / 2 ); //根据当前页 page 的值，计算下一页 page 的值
  }
?>">
<?php
  $page_4 = ($_GET [page]) * 2;                  //获取下一页中输出的第一条记录的值
  if ($page_4 <= 0) {
        $page_4 = 2;                             //如果$page_4 的值小于等于 0，则输出 2
  }
  if ($page_4 == $page_count * 2) {              //如果$page_4 的值等于总记录数的 2 倍
        $page_4 = $page_count * 2 - 1;           //则输出总记录数的 2 倍减 1 的值
  }
$query_4 = mysql_query ( "select * from tb_book limit $page_4,1" );  //以变量 page_4 的值为条件，
执行查询语句
  $myrow_4 = mysql_fetch_array ( $query_4 );
  echo $myrow_4 [bookname];                      //输出指定的数据
?>
          </a></td>
```

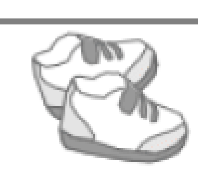
最后，实现以 10 页为一个单位，实现上 10 页和下 10 页的跳转，并且可以在当前页显示的 10 个超链接中任意跳转。其关键代码如下：

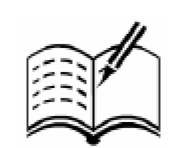
```
<td width="129" align="center" bgcolor="#FFFFFF"><span class="STYLE2"> 分页：
<?php
  $next = $_GET [link_type] * 10;                //定义变量，以 10 页为一个单位
```





```
$n = $link_type - 1;                          //上 10 页
$m = $link_type + 1;                          //下 10 页
$prev_page = $_GET [page] - 10;               //当前页的上 10 页
if ($_GET [link_type] == 0) {                 //如果当前页的值为 0，则输出首页图标
    echo "<img src=\"images/02.jpg\" width=\"8\" height=\"9\" border=\"0\" title=\"首页\">";
} else {                                      //否则，输出首页超链接，并为上 10 页设置链接
    echo    "<a    href='index.php?vv=0&link_type=0&page=1'><img    src=\"images/02.jpg\"
width=\"8\" border=\"0\" height=\"9\" border=\"0\" title=\"首页\"></a> ";
    $ccc = $vv - 10;        //变量$vv 是当前 10 页的起始位置，$ccc 是上一个 10 页的起始位置
    echo     "<a      href='index.php?vv=$ccc&link_type=$n&page=$prev_page'><img
src=\"images/01.jpg\" width=\"8\" height=\"9\" border=\"0\" title=\"上十页\"></a>";
}
?>
<?php
for($j = 1; $j <= $page_count; $j ++) {  //应用 for 循环，输出当前页的 10 个超链接
    $pnext = $next + $j;                 //定义当前页输出的页码值
    if ($mm == 10) {                     //判断当$mm 的值等于 10 时跳出循环
        break;
    }
    if ($mm > $page_count) {             //判断如果变量 mm 的值大于总的记录数，跳出循环
        break;
    }
    if ($page_count - $vv < 10) {        //如果总的记录数减去当前 10 页的值小于 10
if ($mm >= $page_count - $vv) {//判断当变量 mm 的值大于总的记录数减去当前 10 页的值时，
跳出循环
            break;
        }
    }
?>
<?php
    echo "<a href='index.php?vv=$vv&link_type=$link_type&page=$pnext'> $pnext </a>";
    $mm = $mm + 1;                       //变量 mm 的值增加
}
?>
    </span></td>
        <td width="28" align="center" bgcolor="#FFFFFF"><span class="STYLE2">
<?php
    $vv = $vv + $mm;                     //定义下 10 页的起始位置
    if ($page_count - $vv <= 0) {//判断当总记录数小于等于下 10 页的起始位置时，输出尾页
图标
        echo "<img src=\"images/03.jpg\" width=\"8\" height=\"9\" border=\"0\" title=\"尾页\">";
    } else {                             //否则输出下 10 页的超链接
        echo "<a href='index.php?vv=$vv&link_type=$m&page=$pnext'><img src=\"images/03.
jpg\" width=\"8\" height=\"9\" border=\"0\" title=\"下十页\"></a>";
    }
    if ($message_count == 0) {                  //判断当总记录数等于 0 时，输出没有记录
        echo "没有记录!";
    }
?>
    </span></td>
```





技术要点

上下分页以当前页码为基础，实现上一页和下一页的跳转，并且输出上一页的最后一条数据和下一页的第一条数据的内容。

在输出数据库中的数据时应用分栏方法，通过一个 1 行 2 列的表格来输出数据库中的数据。其实现的原理是：

（1）在 while 语句之外，定义一个变量$number=0。

（2）在 while 语句之中，通过 if 语句判断取模（$number % 2）的值是否等于 0，如果等于 0 则是行的开始（执行<tr>）。

（3）在 while 语句之中，通过 if 语句判断取模（$number % 2）的值是否等于 0，如果不等于 0 则是行的结束（执行<\tr>）。

（4）在 while 语句之外，执行$number++。

实例 108　通过 PHP 面向对象实现数据分页

（实例位置：配套资源\SL\06\108　视频位置：配套资源\SP\06\108）

实例说明

实例 107 介绍的是通过 PHP 的面向过程的编程方式实现数据的分页输出，本实例介绍通过面向对象的编程方式实现数据的分页输出。运行结果如图 6.13 所示。

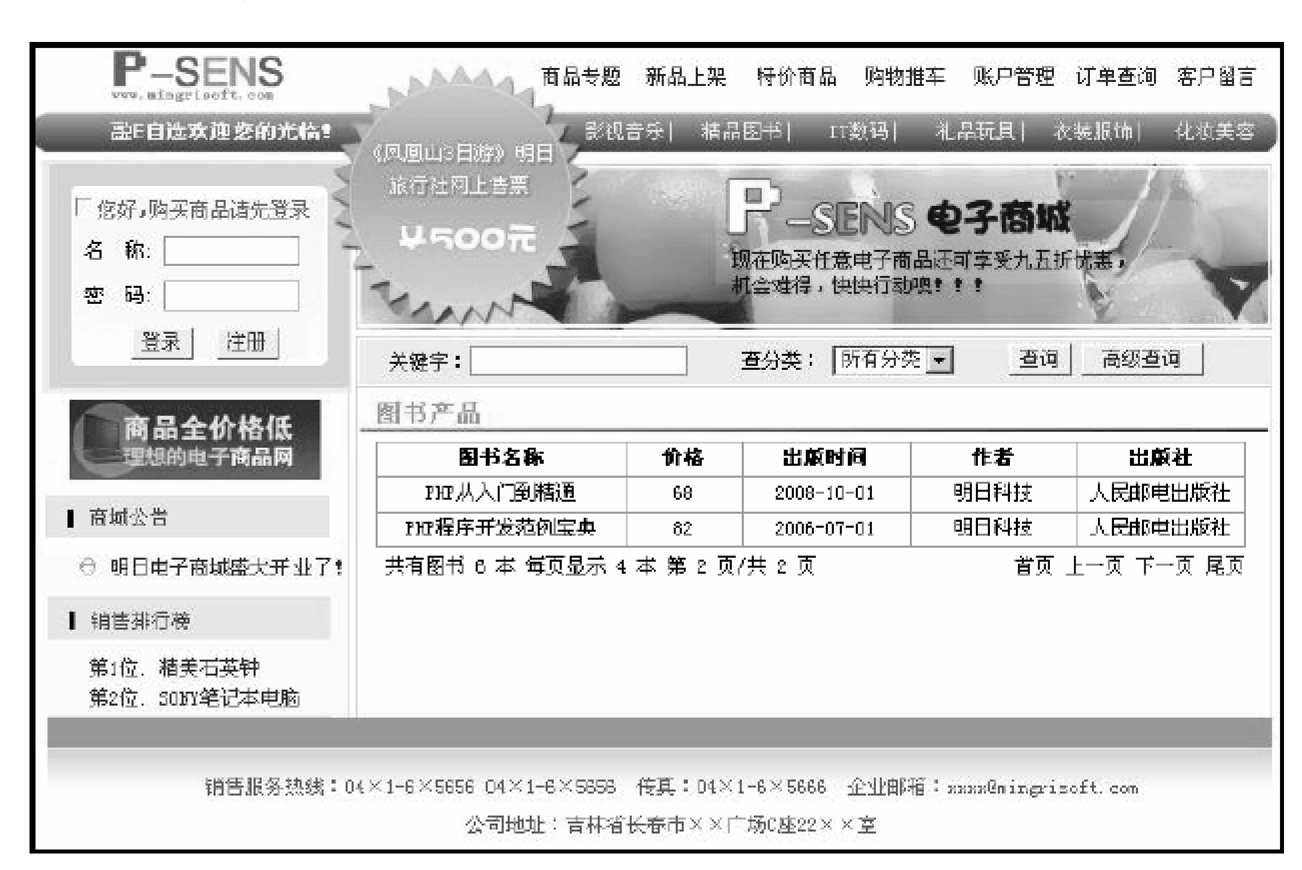


图 6.13　分页输出数据库中数据

在本实例中应用面向对象编程，将分页方法定义到一个类文件中，通过对这个类的实例化操作，完成数据库中数据的分页输出。

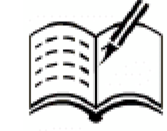
实现过程

具体步骤如下：

（1）创建 page 类，定义分页方法和分页超链接的方法，详细代码可参考技术要点中的内容。

（2）实例化 page 类，通过 listInfo()方法和 toPage()方法完成分页输出数据库中数据的





操作，其关键代码如下：

```
<?php
 $obj=new page(4,$_GET["page"]);                    //实例化 page 类
 $obj->listInfo();                                  //执行分页方法
 $obj->toPage();                                    //执行分页超链接的方法
?>
```

技术要点

面向对象的分页关键就是分页类的创建和类的实例化。这里使用的是分页类 page，在该类中通过 listInfo()方法完成数据的分页操作；通过 toPage()方法完成数据分页超链接的创建。page 类的关键代码如下：

```
<?php
 class page{
     private $pagesize;                             //定义成员变量
  private $page;                                    //定义成员变量
  private $pagecount;                               //定义成员变量
  private $total;                                   //定义成员变量
  private $conn;                                    //定义成员变量
  public function __construct($pagesize,$page){     //定义构造方法，获取参数传递的方法
      $this->pagesize=$pagesize;                    //设置成员变量的值
      $this->page=$page;                            //设置成员变量的值
  }
  public function listInfo(){                       //声明方法
      if($this->page=="" || !is_numeric($this->page)){   //如果分页变量的值为空
          $this->page=1;                            //则为变量赋值 1
      }
      $this->conn=mysql_connect("localhost","root","111");  //连接数据库服务器
      mysql_select_db("db_database06",$this->conn);    //连接数据库
          mysql_query("set names gb2312");          //设置编码格式
      $sql=mysql_query("select count(*) as total from tb_book",$this->conn);       //执行查询
      $myrow=mysql_fetch_array($sql);
      $this->total=$myrow[total];                   //获取查询结果
      if($this->total==0){                       //判断如果查询结果为 0，则输出如下内容
          echo "<table width=520 height=20 border=0 align=center cellpadding=0 cellspacing=0>"+
                   "<tr>"+
                    "<td><div align=center>暂无图书信息！</div></td>"+
                   "</tr>"+
                  "</table>";
      }else{                                                    //否则
      if(($this->total % $this->pagesize)==0){//判断如果总的记录数除以每页显示的记录数等于 0
           $this->pagecount=intval($this->total/$this->pagesize); //则为变量 pagecount 赋值
        }else{
           if($this->total<=$this->pagesize){//如果查询结果小于等于每页记录数，那么为变量赋值为 1
              $this->pagecount=1;
           }else{
               $this->pagecount=ceil($this->total/$this->pagesize);      //否则输出变量值
           }
```




```
                    }
                    $sql=mysql_query("select * from tb_book order by id desc limit ".$this->pagesize*($this->
page-1).",$this->pagesize",$this->conn);
                     ?>
                 …//省略了部分代码
                 <?php
                   while($myrow=mysql_fetch_array($sql)){
                    ?>
                    <tr>
                     <td height="20" bgcolor="#FFFFFF"><div align="center"><?php echo $myrow[bookname];?>
</div></td>
                       <td height="20" bgcolor="#FFFFFF"><div align="center"><?php echo $myrow[price];?>
</div></td>
                      <td height="20" bgcolor="#FFFFFF"><div align="center"><?php echo $myrow[issuDate];?>
</div></td>
                       <td height="20" bgcolor="#FFFFFF"><div align="center"><?php echo $myrow[maker];?>
</div></td>
                      <td height="20" bgcolor="#FFFFFF"><div align="center"><?php echo $myrow[publisher];?>
</div></td>
                      </tr>
                     <?php
                       }
                    echo "</table>";
               }
              }
             public function toPage(){                    //声明方法，创建分页的超链接
             ?>
                 <table width="520" height="24" border="0" align="center" cellpadding="0" cellspacing="0">
                      <tr>
                       <td width="342"> 共有图书 <?php echo $this->total;?> 本 每
页显示 <?php echo $this->pagesize;?> 本 第 <?php echo $this->page;?> 页/
共 <?php echo $this->pagecount;?> 页</td>
                       <td width="362"><div align="right">
                           <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=1">首页</a>
                              <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
                               if($this->page>1)
                                echo $this->page-1;
                               else
                                  echo 1;
                            ?>">上一页</a>
                            <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
                               if($this->page<$this->pagecount-1)
                                  echo $this->page+1;
                               else
                                   echo $this->pagecount;
                            ?>">下一页</a>
                          <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php echo $this->
pagecount;?>">尾页</a>
                     </div></td>
```





```
            </tr>
        </table>
    <?php
    }
        public function __destruct(){                    //声明方法关闭数据库
        mysql_close($this->conn);
      }
    }
?>
```

实例 109　查询结果不显示重复记录

（实例位置：配套资源\SL\06\109）

实例说明

在某些管理系统中所录入的数据包含重复记录是不可避免的，为了在前台显示时不显示重复的记录可以通过 SQL 语言的关键字 distinct 实现。运行本实例，如图 6.14 所示，图书信息中不显示价格相同的信息。



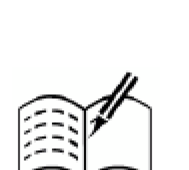
图 6.14　去掉重复项输出

实现过程

创建脚本文件 index.php，启动 MySQL 服务器，连接 MySQL 数据库。单击按钮显示信息。核心代码如下：

```
<?php
    if($_POST[submit]){                                          //单击按钮
    if($_POST[text] != "" || $_POST[text] != "输入图书价格"){    //判断文本框信息
        if(preg_match("/[^a-zA-Z_]/",$_POST[text])){             //正则表达式验证信息
                                                                 //返回结果集
        $rs = mysql_query("select distinct * from tb_demo031 where price = '$_POST[text]元'");
?>
  <table  width="580px"><tr><td  background="pic/head.JPG">ID</td><td  background="pic/head.JPG">
书名</td><td background="pic/head.JPG">价格</td><td background="pic/head.JPG">日期</td></tr>
<?php
            while($rst = mysql_fetch_row($rs)){                  //循环输出结果
?>
  <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php
echo  $rst[1];?></td><td  background="pic/head.JPG"><?php  echo  $rst[2];?></td><td  background="pic/head.
JPG"><?php echo $rst[3];?></td></tr>
```





```
<?php
                }
            }else{
                echo "<script>alert('输入不是一个数值');</script>";        //JavaScript 提示
            }
        }else{
            echo "<script>alert('请输入信息');</script>";                   //JavaScript 提示
        }
    }
?>
```

技术要点

SQL 语句中实现查询结果中不显示重复记录可以通过关键字 distinct 实现，该关键字使用的格式如下：

```
select distinct  字段名  from  表名  where 查询条件
```

distinct 关键字可以在查询结果中去除重复行，与其对应的还有关键字 all，在 SQL 语句中加入关键字 all，表示显示所有的记录。如果 SQL 语句中这两个关键字都不加则查询结果中将显示所有的记录。

实例 110　Delete 语句删除图书信息

（实例位置：配套资源\SL\06\110）

实例说明

数据库可以长期储存数据，但是对于一些已经过期不再使用的数据或者冗余数据信息，如果不及时删除，会影响到 MySQL 服务器的运行效率。所以删除不再使用的数据是十分必要的。本实例通过 Delete 语句实现删除图书信息，运行结果如图 6.15 所示。



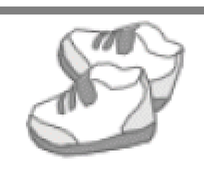
图 6.15　删除图书信息

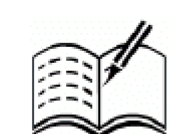
实现过程

具体步骤如下：

（1）创建 in.php 文件，通过面向对象的知识编写数据库连接类。

（2）创建 index.php 文件。首先，创建 form 表单，将删除数据的 ID 提交到本页。然后，包含数据库连接类并实例化对象，判断表单中提交数据是否符合要求，如果符合要求，





则执行 Delete 语句。最后，循环输出数据表中信息，核心代码如下：

```
<?php
  include("in.php");                                                   //包含数据库连接类
  new mysql("localhost","root","111","db_database06");                 //实例化对象
  if($_POST[sub]){                                                     //单击“删除”按钮
      if(preg_match("/\d/",$_POST[na])){                           //首先验证文本框是否为数字
          if(mysql_query("delete from tb_demo029 where id='$_POST[na]'")){//执行删除操作
              echo "<script>alert('删除成功');</script>";               //输出提示
          }else{
              echo "<script>alert('您输入的商品 ID 不在范围内');</script>";
          }
      }else{
          echo "<script>alert('您输入的图书 ID 不是数字');</script>";
      }
  }
  $rs = mysql_query("select * from tb_demo029");                       //查询操作输出信息
  echo "<table width='580px'><td>ID</td><td>书名</td><td>价格</td><td>日期</td></tr>";
  while($rst = mysql_fetch_row($rs)){
      echo   "<td>".$rst[0]."</td><td   bgcolor='#FFFFFF'>".$rst[1]."</td><td>".$rst[2]."</td><td
bgcolor='#FFFFFF'>".$rst[3]."</td></tr>";
  }
  echo "</table>";
?>
```

技术要点

Delete 语句删除表中的数据通常有两种形式：一是可以一次从表中删除多条数据；二是只删除一条记录。其语法格式如下：

```
DELETE [FROM]
{table_name | view_name}
[WHERE search_conditions]
```

参数说明：

☑ table_name：指定要删除数据的数据表的名称。

☑ view_name：用于要删除数据的视图。

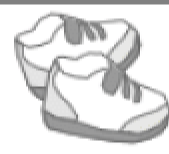
☑ search_conditions：使用搜索条件来限定要删除的数据行。

实例 111　对统计结果进行排序

（实例位置：配套资源\SL\06\111）

实例说明

在商品信息管理系统中，经常需要对商品的价格、数量进行求和，然后根据所求得的值对商品进行排序，从而实现商品的排序输出。运行本实例，如图 6.16 所示，将单一类型商品求和并按商品数量和商品单价排序。


对统计结果进行排序

排序

ID	商品	数量	价格	日期	总价值
3	电风扇	20件	700元	2010-07-21	14000元
2	微波炉	15件	1300元	2010-07-21	19500元
1	电视	10件	2500元	2010-07-21	25000元
4	全自动洗衣机	10件	3500元	2010-07-21	35000元
6	笔记本电脑	8件	3000元	2010-07-21	24000元
5	空调	3件	5000元	2010-07-21	15000元

图 6.16　对统计结果进行排序

实现过程

创建脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“排序”按钮时显示运行结果内容如图 6.16 所示。

```
<?php
  $conn = mysql_connect("localhost","root","111") or die("connect mysql false");  //连接 MySQL
  mysql_select_db("db_database06",$conn) or die ("connect database false");//连接 MySQL 数据库
  mysql_query("set names utf8");                                    //设置编码格式
  if($_POST[sub]){                                                  //单击按钮
      $rs = mysql_query("select * from tb_demo054 order by mount desc,price asc");  //返回结果集
?>
  <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">
商品</td><td background="pic/head.JPG">数量</td><td background="pic/head.JPG">价格</td><td background=
"pic/head.JPG">日期</td><td background="pic/head.JPG">总价值</td></tr>
<?php
      while($rst = mysql_fetch_row($rs)){                           //循环输出结果
?>
  <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG">
<?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?>件</td><td background="pic/
head.JPG"><?php echo $rst[3];?>元</td><td background="pic/head.JPG"><?php echo $rst[4];?><td background=
"pic/head.JPG"><?php echo $rst[2]*$rst[3];?>元</td></td></tr>
<?php
      }
  }
?>
```

技术要点

SQL 语言中实现对某字段所有数据求和可以通过函数 sum()实现，该函数的使用格式如下：

```
sum([all]字段名) 或 sum([distinct]字段名)
```

sum 中的参数如果为 all 加字段名则表示对该字段中的所有记录进行求和，包括重复记录；如果 sum 中的参数为 distinct 加字段名则表示对该字段的所有不重复记录进行求和，在 select 语句中使用 sum()函数的格式如下：

```
select sum(字段名) as 新字段名 from 表名 where 查询条件
```

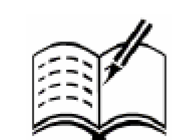


本实例实现对统计结果排序的 SQL 语句代码如下：

```
select * from tb_demo054 order by mount desc,price asc;
```

实例 112　使用 select 子句进行多表查询

（实例位置：配套资源\SL\06\112　视频位置：配套资源\SP\06\112）



实例说明

在实际项目开发过程中，经常需要将不同的信息存储在不同的表中，表与表之间通过某字段相互关联，从而使表的指针形成一种连动关系进而可以通过 SQL 语言的 select 语句实现多表查询。运行本实例，如图 6.17 所示，在图中文本框中输入要查询成绩的学生学号，单击“查询”按钮即可将学生信息和该学生的各科成绩查找出来。

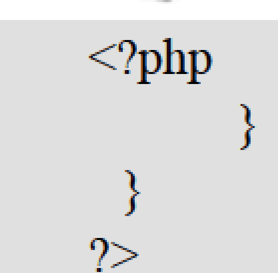
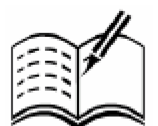


图 6.17　多表查询

实现过程

创建 index.php 文件，当用户在文本框中输入数据并单击“查询”按钮时显示如图 6.17 所示的内容。核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("Connect MySQL false");//连接 MySQL
  mysql_select_db("db_database06",$conn);                    //连接数据库
  mysql_query("SET NAMES UTF8");                             //设置编码格式
  if($_POST[sub]){                                           //单击按钮
                                                             //返回结果集
        $rs = mysql_query("select a.id,a.name,a.address,a.date,b.math,b.english,b.chinese from tb_demo065_tel as b,tb_demo065 as a where a.id=b.id and a.id='$_POST[text]'");
?>
  <table width="580px"><tr><td background="pic/head.JPG">学生 ID</td><td background="pic/head.JPG">学生姓名</td><td background="pic/head.JPG">家庭住址</td><td background="pic/head.JPG">入学日期</td><td background="pic/head.JPG">数学考试分数</td><td background="pic/head.JPG">英语考试分数</td><td background="pic/head.JPG">语文考试分数</td></tr>
<?php
        while($rst = mysql_fetch_row($rs)){                  //循环输出结果
?>
  <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background="pic/head.JPG"><?php echo $rst[3];?></td><td background="pic/head.JPG"><?php echo $rst[4];?></td><td background="pic/head.JPG"><?php echo $rst[5];?></td><td background="pic/head.JPG"><?php echo $rst[6];?></td></tr>
```

```
<?php
        }
  }
?>
```

技术要点

SQL 语言中通过 where 子句实现多表查询，所要查找的字段名最好用“表名.字段名”表示，这样可以防止因表之间字段重名而造成无法获知该字段属于哪个表，在 where 子句中多个表之间所形成的连动关系应按如下形式书写：

```
表 1.字段=表 2.字段  and  其他查询条件
```

综上所述，实现多表查询，应按如下形式：

```
select  字段名  from  表 1,表 2… where  表 1.字段=表 2.字段  and  其他查询条件
```

实例 113　合并多个结果集

（实例位置：配套资源\SL\06\113）

实例说明

SQL 语言中使用关键字 union 可以将多张表的信息合并输出。图 6.18 为数据库中无关联的两张信息表，运行本实例，单击“合并”按钮即可将这两张表的信息合并输出，如图 6.19 所示。

			uid	name	price	date
□	✎	✕	1	电视	15100	2010-07-24
□	✎	✕	2	空调	5000	2010-07-24
□	✎	✕	3	全自动洗衣机	3500	2010-07-24
□	✎	✕	4	热水器	1200	2010-07-24

			id	name	pwd
□	✎	✕	1	小杨	1081
□	✎	✕	2	小潘	1026
□	✎	✕	3	小刘	1015
□	✎	✕	4	小黄	1077

图 6.18　两张无关联表



ID	名称	说明
1	电视	15100
2	空调	5000
3	全自动洗衣机	3500
4	热水器	1200
1	小杨	1081
2	小潘	1026
3	小刘	1015
4	小黄	1077

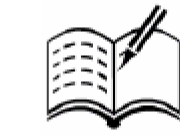
图 6.19　合并多个结果集

实现过程

创建脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“合并”按钮时显示如图 6.19 所示的内容。代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("Connect MySQL false");//连接 MySQL
```






```
        mysql_select_db("db_database06",$conn);                    //连接数据库
        mysql_query("SET NAMES UTF8");                             //设置编码集
        if($_GET[id] == "1"){                                      //当 id 等于 1 时
      //返回结果集
      $rs=mysql_query("select uid,name,price from tb_demo067_tel union select id,name,pwd from tb_demo067") or die
("false");
      ?>
        <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">
名称</td><td background="pic/head.JPG">说明</td></tr>
      <?php
            while($rst = mysql_fetch_array($rs)){                  //循环输出
      ?>
        <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG">
<?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td></tr>
      <?php
            }
        }
      ?>
```

技术要点

SQL 语言中可以通过关键字 union 或 all 将多个 select 语句的查询结果合并输出，这两个关键字的使用说明如下：

（1）union：利用该关键字可以将多个 select 语句的查询结果合并输出，并删除重复行。

（2）all：利用该关键字也是将多个 select 语句的查询结果合并输出但不删除重复行。

在使用 union 或 all 关键字将多张表合并输出时，查询结果必须具有相同的结构并且数据类型必须兼容。

实例 114　简单的嵌套查询

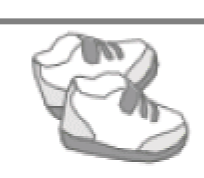
（实例位置：配套资源\SL\06\114）

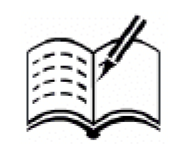
实例说明

在数据库系统开发过程中嵌套查询被广泛地应用，本实例将通过嵌套查询实现按指定的学生学号查询学生的成绩信息。运行本实例，如图 6.20 所示，首先在图中的文本框中输入要查询的学生的学号，然后单击“查询”按钮即可将该学号所对应的学生信息查找出来。



图 6.20　嵌套查询





实现过程

创建脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库，在文本框中输入学生 ID，单击“查询”按钮，显示如图 6.20 所示内容。核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("Connect MySQL false");//连接 MySQL
  mysql_select_db("db_database06",$conn);                    //连接 MySQL 数据库
  mysql_query("SET NAMES UTF8");                             //设置编码格式
  if($_POST[sub]){                                           //单击按钮
       //返回结果集
       $rs = mysql_query("select id,name,sex,date from tb_demo068 where id in(select id from tb_demo068 where id=$_POST[text])") or die ("false");
?>
  <table width="580px"><tr><td background="pic/head.JPG">学生 ID</td><td background="pic/head.JPG">学生姓名</td><td background="pic/head.JPG">学生性别</td><td background="pic/head.JPG">入学时间</td></tr>
<?php
       while($rst = mysql_fetch_array($rs)){                 //循环输出结果
?>
  <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background="pic/head.JPG"><?php echo $rst[3];?></td></tr>
<?php
       }
  }
?>
```

技术要点

本实例是通过一个嵌套子查询实现的。子查询是一个 select 查询，返回单个值且嵌套在 select、insert、update 和 delete 语句或其他查询语句中。任何可以使用表达式的地方都可以使用子查询。

子查询也称内部查询或内部连接，包含子查询的语句称为外部查询或外部选择。本实例中实现嵌套查询的代码如下：

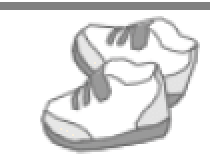
```
select id,name,sex,date from tb_demo068 where id in(select id from tb_demo068 where id=$_POST[text]);
```

实例 115　用 in 查询表中的记录信息

（实例位置：配套资源\SL\06\115）

实例说明

SQL 语言中可以利用关键字 in 来查询表中满足一定条件的记录，运行本实例，如图 6.21 所示，首先在图中文本框中输入要查询的图书名称，单击“查询”按钮即可将与用





户输入的查询关键字相匹配的图书查找出来。

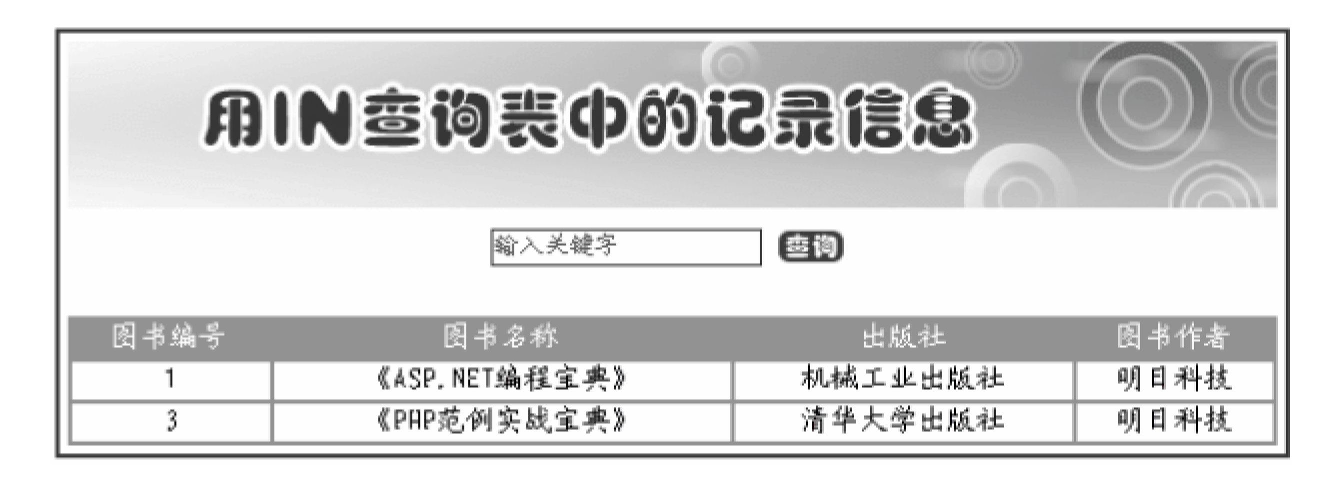


图 6.21 用 in 查询表中记录

实现过程

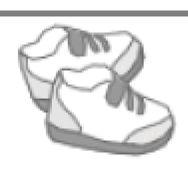
创建脚本文件 index.php。编写网页框架，启动 MySQL 服务，连接 MySQL 数据库。编写<form>表单，当单击“查询”按钮时显示如图 6.21 所示的内容。核心代码如下：

```
<?php
if($_POST[sub]){                                          //单击“查询”按钮
    if($_POST[te] == "" || $_POST[te] == "输入关键字"){     //对文本框信息进行判断
        echo "<script>alert('请输入内容');</script>";        //JavaScript 提示
    }else{
        //sql 语句
            $sql = "select * from tb_demo084 where name in (select name from tb_demo084
where name like '%$_POST[te]%' ) ";
            $rs = mysql_query($sql);                      //返回结果集
?>
            <table width="580px"><tr><td background="pic/head.JPG" align="center">图书
编号</td><td background="pic/head.JPG" align="center">图书名称</td><td background="pic/head.JPG"
align="center">出版社</td><td background="pic/head.JPG" align="center">图书作者</td></tr>
<?php
            while($rst = mysql_fetch_row($rs)){           //循环输出结果
?>
            <tr><td background="pic/head.JPG" align="center"><?php echo $rst[0];?></td>
<td background="pic/head.JPG" align="center"><?php echo $rst[1];?></td><td background="pic/head.JPG"
align="center"><?php echo $rst[2];?></td><td background="pic/head.JPG" align="center"><?php echo $rst[3];?>
</td></tr>
<?php
            }
        }
    }
?>
```

技术要点

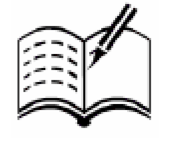
本实例首先通过 in 子查询查找出所有与用户输入的关键字相匹配的图书名称，然后通过外部查询查找出所有图书的详细信息，代码如下：

```
select * from tb_demo084 where name in (select name from tb_demo084 where name like '%$_
POST[te]%';
```




实例 116　使用聚集函数 sum()对学生成绩进行汇总

（实例位置：配套资源\SL\06\116）


Note

实例说明

在学生成绩管理系统中，经常需要求出班级所有学生的各科成绩总和，从而求出所有学生的各科平均成绩。运行本实例，如图 6.22 所示，单击图中的“对学生成绩汇总”按钮即可将班级的各科学生成绩统计出来。

使用聚集函数对学生成绩进行汇总

对学生成绩汇总

ID	学生姓名	语文成绩	数学成绩	英语成绩	总成绩
2	小潘	96	95	94	285
1	小杨	97	96	80	273
3	小刘	90	90	90	270
4	小黄	60	60	60	180

图 6.22　对学生成绩汇总

实现过程

编写脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库，当单击“对学生成绩汇总”按钮时，显示如图 6.22 所示信息。核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("connect mysql false"); //连接 MySQL
mysql_select_db("db_database06",$conn) or die ("connect database false");  //连接 MySQL 数据库
  mysql_query("set names utf8");                                  //设置编码格式
  if($_GET[page] == "1"){                                         //page 等于 1 时
  $rss = mysql_query("select *,sum(chinese+math+english) as a from tb_demo059 group by name
order by a desc ");                                               //返回结果集
?>
  <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">
学生姓名</td><td background="pic/head.JPG">语文成绩</td><td background="pic/head.JPG">数学成绩
</td><td background="pic/head.JPG">英语成绩</td><td background="pic/head.JPG">总成绩</td></tr>
<?php
      while($rst = mysql_fetch_row($rss)){                        //循环输出结果
?>
      <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.
JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background=
"pic/head.JPG"><?php echo $rst[3];?></td><td background="pic/head.JPG"><?php echo $rst[4];?></td><td
background="pic/head.JPG"><?php echo $rst[5]?></td></tr>
<?php
    }
  }
?>
```



技术要点

SQL 语句中，实现对某字段的所有记录进行求和可以通过函数 sum()实现，该函数的使用方法如下：

```
sum([all | distinct] expression)
```

参数说明：

☑ all：表示对指定字段的所有值进行聚集函数运算，all 为默认值，如果不加参数 all 或 distinct 则表示对指定字段的所有记录进行聚集运算。

☑ distinct：表示对指定字段的所有非重复记录进行求和。

☑ expression：是精确数字或近似数字数据类型分类（bit 数据类型除外）的表达式。

sum()函数的返回类型如表 6.3 所示。

表 6.3　sum()函数的返回类型

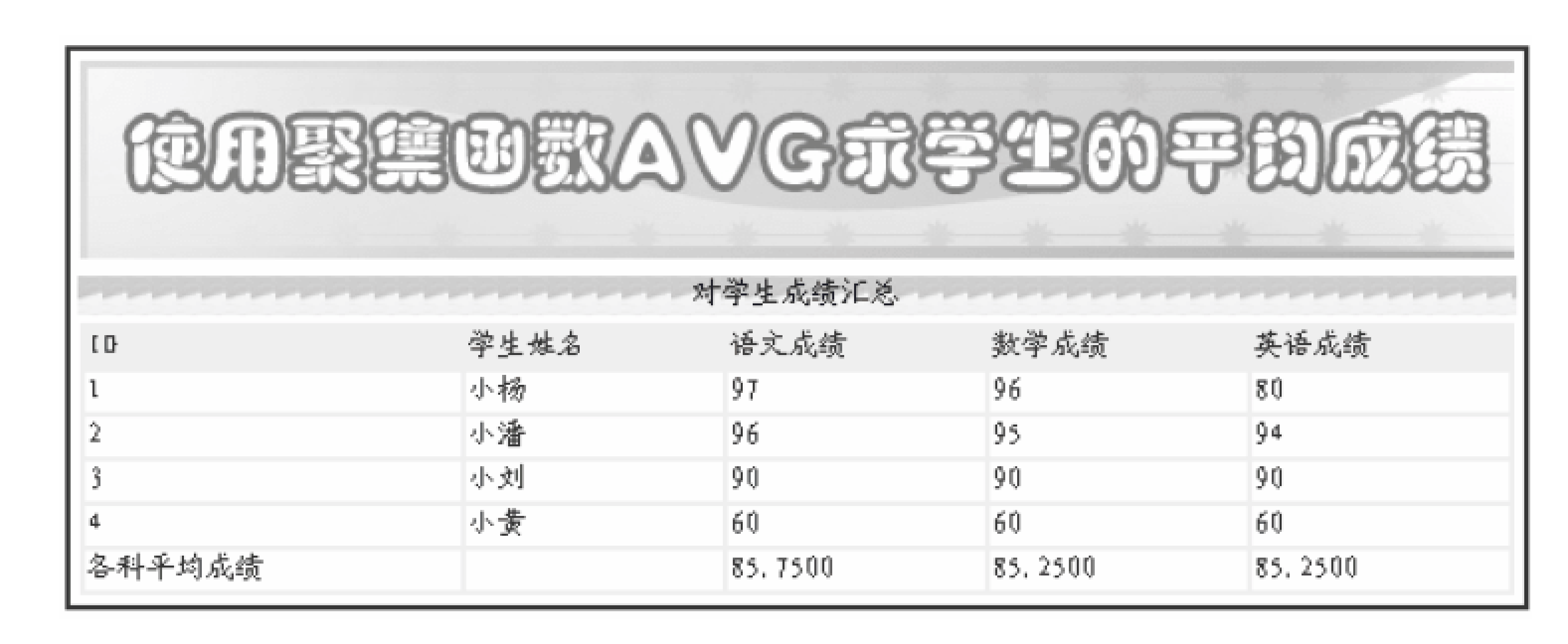
表达式结果	返 回 类 型
整数分类	int
decimal 分类	decimal(38,s) 除以 decimal(10,0)
money 和 smallmoney 分类	money
float 和 real 分类	float

实例 117　使用聚集函数 avg 求学生的平均成绩

（实例位置：配套资源\SL\06\117）

实例说明

获取统计数据平均值的方法有多种，例如，可以先求得所有数据的总和，然后再除以总个数，而 SQL 语言中提供了专门求平均数的函数 avg()，本实例将利用该函数求得学生成绩表中各科学生成绩的平均值。运行本实例，如图 6.23 所示，单击图中 “对学生成绩汇总”的按钮即可将各科的平均成绩显示出来。



ID	学生姓名	语文成绩	数学成绩	英语成绩
1	小杨	97	96	80
2	小潘	96	95	94
3	小刘	90	90	90
4	小黄	60	60	60
各科平均成绩		85.7500	85.2500	85.2500

图 6.23　取得学生的各科平均成绩

实现过程

创建脚本文件 index.php。当单击“对学生成绩汇总”按钮时显示如图 6.23 所示的内



容，实现此功能的核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("connect mysql false"); //连接 MySQL
  mysql_select_db("db_database06",$conn) or die ("connect database false");//连接 MySQL 数据库
  mysql_query("set names utf8");                                       //设置编码集
  if($_GET[page] == "1"){                                              //如果 page 等于 1
        $rss = mysql_query("select * from tb_demo059");                //返回结果集
?>
  <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">
学生姓名</td><td background="pic/head.JPG">语文成绩</td><td background="pic/head.JPG">数学成绩
</td><td background="pic/head.JPG">英语成绩</td></tr>
<?php
            while($rst = mysql_fetch_row($rss)){                       //循环输出结果
?>
            <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/
head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background=
"pic/head.JPG"><?php echo $rst[3];?></td><td background="pic/head.JPG"><?php echo $rst[4];?></td></tr>
<?php
        }
        $rts = mysql_query("select avg(chinese),avg(math),avg(english) from tb_demo059");//返回结
果集
        $rsth = mysql_fetch_row($rts);
?>
  <tr><td background="pic/head.JPG">各科平均成绩</td><td background="pic/head.JPG"></td><td
background="pic/head.JPG"><?php echo $rsth[0];?></td><td background="pic/head.JPG"><?php echo $rsth[1];?>
</td><td background="pic/head.JPG"><?php echo $rsth[1];?></td></tr>
<?php
  }
?>
```

技术要点

获取某字段的所有记录的平均成绩可以通过函数 avg()实现，该函数的使用方法如下：

```
avg([all | distinct] expression)
```

本实例实现利用 avg 函数实现显示平均成绩的 SQL 语句代码如下：

```
select avg(chinese),avg(math),avg(english) from tb_demo059;
```

实例 118　使用聚集函数 min()求利润最少的商品

（实例位置：配套资源\SL\06\118）

实例说明

在销售管理系统中，商家会根据不同商品的销售利润来确定该商品的进货数量。运行本实例，如图 6.24 所示，单击图中的“获取利润最小的商品信息”按钮即可将销售利润最



小的商品信息显示在图中最后一行。

使用聚集函数MIN求利润最少的商品



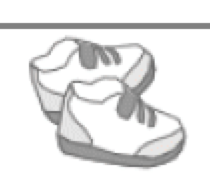
ID	商品名称	进货价格/元	销售价格/元	进货数量/件	进货日期
1	电视	1500	1600	10	2010-07-22
2	冰箱	3100	3300	5	2010-07-22
3	空调	4500	4900	1	2010-07-22
4	笔记本电脑	3000	3500	8	2010-07-22
利润最小的商品	电视	100			

图 6.24　利润最小的商品输出

实现过程

创建脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“获取利润最小的商品信息”按钮时显示如图 6.24 所示的内容。核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("connect mysql false"); //连接 MySQL
  mysql_select_db("db_database06",$conn) or die ("connect database false");//连接 MySQL 数据库
  mysql_query("set names utf8");                                        //设置编码格式
  $rss = mysql_query("select * from tb_demo061");                       //返回结果集
?>
  <table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">商品名称</td><td background="pic/head.JPG">进货价格/元</td><td background="pic/head.JPG">销售价格/元</td><td background="pic/head.JPG">进货数量/件</td><td background="pic/head.JPG">进货日期</td></tr>
<?php
        while($rst = mysql_fetch_row($rss)){                            //循环输出结果
?>
        <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background="pic/head.JPG"><?php echo $rst[3];?></td><td background="pic/head.JPG"><?php echo $rst[4];?></td><td background="pic/head.JPG"><?php echo $rst[5];?></td></tr>
<?php
        }
  if($_GET[page] == "1"){                                               //当 page 等于 1 时
      $sql = "select min(outshop-inshop), name from tb_demo061 group by date ";
//sql 语句
      $rs = mysql_query($sql);                                          //返回结果集
      $rst = mysql_fetch_row($rs);                                      //输出结果
?>
  <tr><td background="pic/head.JPG">利润最小的商品</td><td background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"></td><td background="pic/head.JPG"></td><td background="pic/head.JPG"></td></tr></table></td></tr></table></td></tr></table>
<?php
  }
?>
```



技术要点

SQL 语言中获取某字段的最小值可以通过函数 min()实现，该函数的使用方法如下：

```
mixed min ( number arg1, number arg2 , number ...)
mixed min ( array numbers , array ...)
```



如果仅有一个参数且为数组，min()函数返回该数组中最小的值。如果第一个参数是整数、字符串或浮点数，则至少需要两个参数而 min()函数会返回这些值中最小的一个。可以比较无限多个值。

实例 119　使用聚集函数 max()求销售利润最高的商品

（实例位置：配套资源\SL\06\119）

实例说明

商品销售管理系统中，商品的销售利润统计模块是必不可少的。运行本实例，如图 6.25 所示，在图中文本框中输入某月份，单击“查询”按钮即可将销售额最大商品信息显示出来。

使用聚集函数MAX求利润最高的商品

请输入月份：　查询

商品ID	名称	销售额	商店名称	销售日期	销售员
1	钻戒	5570	祥云珠宝行	2010-12-08	张林
2	18k吊坠	320402	祥云珠宝行	2010-12-11	小李
366	PT900耳钉	215456	东方广场	2010-12-05	孙丽
367	18k钻石爱链	3000	东方广场	2010-12-07	刘芳

图 6.25　获取月利润最大的商品信息

实现过程

具体步骤如下：

（1）创建 conn.php 文件，实现与数据库的连接，代码如下：

```
<?php
$conn=mysql_connect("localhost","root","111");                    //连接 MySQL 数据库
mysql_select_db("db_database06",$conn);                           //连接数据库
mysql_query("set names gb2312");                                  //设置编码格式
?>
```

（2）显示商品表中所有商品的销售信息，代码如下：

```
<?php
   include_once("conn.php");                                        //包含数据库连接文件
   $sql=mysql_query("select * from tb_demo062",$conn);              //执行查询操作
   $info=mysql_fetch_array($sql);                                   //返回结果集
   if($info==false) {                                               //判断结果集是否存在
       echo "暂无商品信息!";
    }else{
      do {
   ?>
    <tr>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php   echo $info[id];?></div>
</td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php   echo $info[name];?>
```





```
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[sale];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[shopname];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[xdate];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[sellman];?>
</div></td>
         </tr>
          <?php
          }while($info=mysql_fetch_array($sql));                    //循环输出结果
         }
            ?>
```

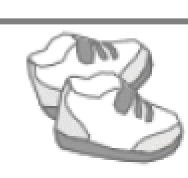
（3）显示月销售额最大商品信息，代码如下：

```
    <?php
    if($_POST[submit]!="") {                                        //通过 POST 方式传递参数
         $yue=$_POST[yue];                                          //将参数值赋给变量
         $sql=mysql_query("select * from tb_demo062 where sale in (select max(sale) from
tb_demo062 where month(xdate)=$yue and year(xdate)=2005)",$conn);   //执行查询操作
         $info=mysql_fetch_array($sql);                             //返回结果集
         if($info==false){                                          //如果结果集不存在
            echo "不存在该月!";
         }else{
       ?>
        <tr>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[id];?></div>
</td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[name];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[sale];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[shopname];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[xdate];?>
</div></td>
            <td height="25" bgcolor="#FFFFFF"><div align="center"><?php  echo $info[sellman];?>
</div></td>
        </tr>
       <?php
          }
         }
      mysql_close($conn);                                           //关闭数据库连接
     ?>
```

技术要点

实现本实例主要应用到 SQL 语言中的 max()函数，该函数的语法格式如下：

```
    max([all | distinct] expression)
```



参数说明：

☑ all：表示对指定字段的所有值进行聚集函数运算，all 为默认值，如果不加参数 all 或 distinct 则表示对指定字段的所有记录进行聚集运算。

☑ distinct：表示对指定字段的所有非重复记录进行求最大值。

☑ expression：是精确数字或近似数字数据类型分类（bit 数据类型除外）的表达式。

实例 120　使用聚集函数 count()求利润大于某值的数据

（实例位置：配套资源\SL\06\120）

实例说明

在 PHP 项目开发过程中，SQL 语言的 count()函数被广泛地应用，例如，在实现记录的分页显示时，需要通过 count()函数求出表中所有记录总和，本实例将讲解 SQL 语言中 count()函数的应用。运行本实例，如图 6.26 所示，首先在图中的文本框中输入某数值，然后单击“查询”按钮即可将所有利润大于该数值的数据求出。

取得大于某值的数据

1600 查询

ID	商品名称	进货价格/元	销售价格/元	进货数量/件	进货日期
1	电视	1500	1600	10	2010-07-22
2	冰箱	3100	3300	5	2010-07-22
3	空调	4500	4900	1	2010-07-22
4	笔记本电脑	3000	3500	8	2010-07-22

共查找到3条

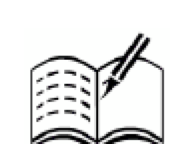
图 6.26　取得大于某值的数据

实现过程

创建脚本文件 index.php，启动 MySQL 服务并连接 MySQL 数据库，当在文本框中输入内容时，单击“查询”按钮，实现如图 6.26 所示的内容。核心代码如下：

```
<?php
        if($_POST[sub]){                                               //单击按钮
            if($_POST[text] != "" || $_POST[text] != "输入查询值"){       //判断文本框信息
                if(preg_match("/\d/",$_POST[text])){              //正则表达式验证信息
 $rs = mysql_query("select count(*) as a from tb_demo061 where $_POST[text] < outshop");//返回
结果集
                    $rst = mysql_fetch_row($rs);                           //输出结果
?>
 <table><tr><td width="580px" align="center">共查找到<?php echo $rst[0];?>条</td></tr></table>
<?php
                }else{
                    echo "<script>alert('请正确输入');</script>";           //JavaScript 提示
                }
            }else{
```






```
                echo "<script>alert('请在文本框中输入内容');</script>";//JavaScript 提示
            }
        }
?>
```

技术要点

统计表中满足一定条件的记录个数，可以通过 SQL 语言的 count()函数实现，该函数的使用格式如下：

```
int count ( mixed array [, int mode])
```

count()函数的参数说明如表 6.4 所示。

表 6.4 count()函数的参数说明

参　数	说　明
array	必选要参数。输入的数组
mode	可选参数。COUNT_RECURSIVE（或 1），如选中此参数，本函数将递归地对数组计数。对计算多维数组的所有单元尤其有用。此参数的默认值为 0

本实例实现取得大于某值的数据的 SQL 语句代码如下：

```
select count(*) as a from tb_demo061 where $_POST[text] < outshop;
```

实例 121 多表联合查询

（实例位置：配套资源\SL\06\121）

实例说明

多表联合查询可以解决连接或简单子查询难于处理的问题。运行本实例，如图 6.27 所示，单击图中的“合并学生信息”按钮即可将“计算机系”和“数学系”的学生信息合并输出。



图 6.27 多表联合查询

实现过程

创建脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“合并学生信息”按钮时显示如图 6.27 所示的内容。核心代码如下：



```
<?php
 if($_GET[id] == "1"){                                    //如果地址栏 id 等于 1
```




```
            //返回结果集
            $rs = mysql_query("select * from tb_demo074_student union select * from tb_demo074_ fasten");
      ?>
        <table  width="580px"><tr><td  background="pic/head.JPG">学生  ID</td><td  background="pic/head.JPG">学生姓名</td><td background="pic/head.JPG">入学时间</td></tr>
      <?php
            while($rst = mysql_fetch_row($rs)){                                    //循环输出结果
      ?>
        <tr><td  background="pic/head.JPG"><?php  echo  $rst[0];?></td><td  background="pic/head.JPG"><?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td></tr>
      <?php
            }
        }
      ?>
```

技术要点

利用 SQL 语言中的关键字 union 可以将不同表中符合条件的数据信息显示在同一列中，union 语句的使用格式如下：

```
    <query specification1> [,query specification2…] union [all] <query specification1> [,query specification2…]
```

参数说明：

☑ <query specification1>：是查询规范或查询表达式。

☑ union：组合多个结果集并将其作为单个结果集返回。

☑ all：在结果集中包含所有的行，包含重复行。

实例 122　left outer join 查询

（实例位置：配套资源\SL\06\122）

实例说明

如果要包含连接的两个表中的不匹配行，可以通过左连接或右连接的方式实现。运行本实例，如图 6.28 所示，单击图中的“查询”按钮即可将员工信息表中所有记录以及员工工资表中与员工 ID 相匹配的记录显示出来。

left outer join查询

查询

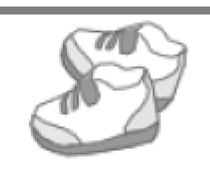
员工姓名	入职时间	员工工资
小杨	2010-07-26	2800
小潘	2010-07-26	2900
小刘	2010-07-26	3000

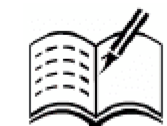
图 6.28　left outer join 查询

实现过程

编写脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“查询”







按钮时显示如图 6.28 所示的内容。核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("Connect MySQL False");
  mysql_select_db("db_database06",$conn);
  mysql_query("SET NAMES UTF8");
  if($_GET['id'] == "1"){
        $rs = mysql_query("select * from tb_demo078_wages right outer join tb_demo078_informaction on
tb_demo078_wages.id=tb_demo078_informaction.id ");
?>
<table width="580px" bgcolor="#F9DABB">
  <tr>
        <td style="color:#FFFFFF" align="center">员工姓名</td>
           <td style="color:#FFFFFF" align="center">入职时间</td>
           <td style="color:#FFFFFF" align="center">员工工资</td>
        </tr>
<?php
  while($rst = mysql_fetch_row($rs)){
?>
  <tr>
        <td bgcolor="#FFFFFF" align="center"><?php echo $rst[3];?></td>
           <td bgcolor="#FFFFFF" align="center"><?php echo $rst[5];?></td>
           <td bgcolor="#FFFFFF" align="center"><?php echo $rst[1];?></td>
      </tr>
<?php
  }
  ?>
</table>
<?php
}
?>
```

技术要点

左连接返回的查询结果包含左表中的所有符合查询条件及右表中所有满足连接条件的行，MySQL 数据库中使用左连接的语法格式如下：

```
select field 1[field2…] from table1 left [outer] join table2 on join_condition [where search_condition]
```

参数说明：

☑ left outer join：表示表之间通过左连接方式相互连接，也可以简写成 left join。

☑ on join_condition：指多表建立连接所使用的连接条件。

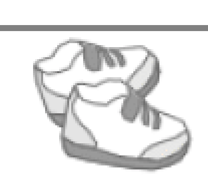
☑ where search_condition：可选参数，用于设置查询条件。

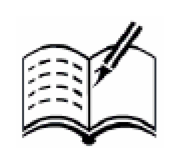
实例 123　right outer join 查询

（实例位置：配套资源\SL\06\123）

实例说明

通过右连接可以将右表中满足查询条件的记录全部显示出来并显示左表中满足连接





条件的记录。运行本实例，如图 6.98 所示，单击图中的“查询”按钮即可将员工信息表中满足连接条件的记录及员工信息表中所有记录全部显示出来。

right outer join查询

查询

员工姓名	员工性别	入职时间	员工工资
小杨	男	小杨	2010-07-26
小潘	男	小潘	2010-07-26
小刘	男	小刘	2010-07-26

图 6.29　right outer join 查询

实现过程

编写脚本文件 index.php，启动 MySQL 服务，连接 MySQL 数据库。当单击“查询”按钮时显示如图 6.29 所示内容。本实例核心代码如下：

```
<?php
  $conn = mysql_connect("localhost","root","111") or die ("Connect MySQL False");
  mysql_select_db("db_database06",$conn);
  mysql_query("SET NAMES UTF8");
  if($_GET[id] == "1"){
        $rs = mysql_query("select * from tb_demo078_wages right outer join tb_demo078_informaction
on tb_demo078_wages.id=tb_demo078_informaction.id ");
?>
  <table width="580px" bgcolor="#3F9DD0"><tr><td style="color:#FFFFFF" align="center">员工姓
名</td><td style="color:#FFFFFF" align="center">员工性别</td><td style="color:#FFFFFF" align="center">
入职时间</td></tr>
<?php
        while($rst = mysql_fetch_row($rs)){
?>
  <tr><td bgcolor="#FFFFFF" align="center"><?php echo $rst[3];?></td><td bgcolor="#FFFFFF"
align="center"><?php echo $rst[4];?></td><td bgcolor="#FFFFFF" align="center"><?php echo $rst[5];?>
</td></tr>
<?php
        }
  }
?>
```

技术要点

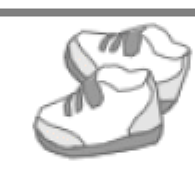
右连接返回的查询结果包含左表中的所有符合连接条件以及右表中所有满足查询条件的行，MySQL 数据库中使用右连接的语法格式如下：

```
select field 1[field2…] from table1 right [outer] join table2 on join_condition [where search_
condition]
```

参数说明：

☑ right outer join：表示表之间通过左连接方式相互连接，也可以简写成 right join。

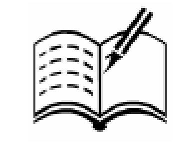
☑ on join_condition：指多表建立连接所使用的连接条件。



☑ where search_condition：可选参数，用于设置查询条件。

实例 124 利用 transform 分析数据

（实例位置：配套资源\SL\06\124）



实例说明

利用 transform 可以动态地按照类别或分组统计出所需要的数据。运行本实例，如图 6.30 所示，单击“统计”按钮即可按月份将不同种类图书的数量统计出来。

利用transform分析数据

统计

书名	2004-4	2004-5	2004-6	2004-7
C	0	1263	200	
Java			30	
QB	958			
VB				1121

图 6.30 利用 transform 分析数据

实现过程

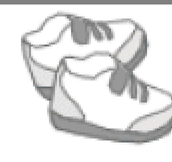
具体步骤如下：

（1）创建 conn.php 文件，实现与数据库的连接，代码如下：

```
<?php
$conn=new com("adodb.connection");          //连接数据库
$connstr="provider=microsoft.jet.oledb.4.0;data source=".realpath("./data/db_database06.mdb");
$conn->open($connstr);
?>
```

（2）执行查询，并显示查询结果，代码如下：

```
<?php
        include("conn.php");                 //包含数据库连接文件
        $rs=new com("adodb.recordset");
                                             //查询结果
        $rs->open("transform sum(number) as total select yyassort from tb_booksort where yyassort in (select yyassort from tb_booksort) group by yyassort pivot analysetime",$conn,3,1);
        if($rs->eof|| $rs->bof){
        ?>
       <tr>
         <td height="25" colspan="8" bgcolor="yellow"><div align="center"></div></td>
       </tr>
       <?php
         } else{
              while(!$rs->eof){
       ?>
```

Note

```
                <tr>
                  <?php
                 for($i=0;$i<=$rs->fields->count-1;$i++){
                ?>
                 <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields($i); echo
iconv('gbk','utf-8',$f->value)?></div></td>
                <?php
                 }
                ?>
                </tr>
                <?php
                     $rs->movenext;
                   }
             }
            ?>
             </table></td>
          </tr>
        </table>
        <?php
         }
        ?>
```

技术要点

本实例中利用 transform 创建交叉表查询，使之能按月份统计不同名称图书的数量。transform 的语法格式如下：

```
transform aggfunction
select statement
pivot pivotfield [in (value1[,value1]…[,valuen])]
```

参数说明：

☑ aggfunction：操作所选数据库的 SQL 聚合函数。

☑ select statement：要执行的 select 语句。

☑ pivotfield：希望用于创建查询结果集中列标题的字段或表达式。

☑ value1...valuen：用于创建列标题的固定值。

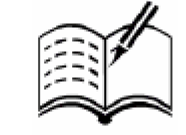
实例 125　使用格式化函数转换查询条件的数据类型

（实例位置：配套资源\SL\06\125）

实例说明

本实例将利用 format()函数实现日期的格式化输出，由于该函数只适用于 Access 数据库，所以本实例将以 Access 作为后台数据库，运行本实例，如图 6.31 所示，单击图中的“转换”按钮即可将内容显示出来，并将日期格式由“xxxx-xx-xx”转换为“xxxx 年 xx 月 xx 日”格式。





格式化函数转换查询条件数据类型

将日期转变为长格式输出 转换

	价格	库存量	类别	进货日期
Visual Basic 6.0数据库开发技术与工程实践	45	121	实例	2004年07月01日
Visual Basic数据库开发实例导航	44	700	实例	2004年07月01日
Visual Basic精彩编程200例	39	300	实例	2004年07月01日
高校课程学练考系列丛书-c语言学练考	30	200	教材	2004年06月01日
Java数据库高级教程	46	30	基础	2004年06月01日
新世纪-计算机基础教育丛书	26	1063	基础	2004年05月01日
C游戏编程从入门到精通	36	200	教材	2004年05月01日
QBASIC程序设计	14	958	教材	2004年04月01日
C语言习题解答	0	0	教材	2004年04月01日

图 6.31　格式化函数转换查询条件数据类型

实现过程

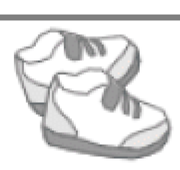
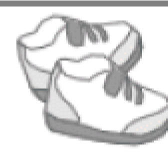
具体步骤如下：

（1）创建 conn.php 文件，实现与数据库的连接，代码如下：

```
<?php
$conn=new com("adodb.connection");                    //连接 Access 数据库
$connstr="provider=microsoft.jet.oledb.4.0;data source=".realpath("./data/db_database06.mdb");
$conn->open($connstr);
?>
```

（2）执行查询，并将日期转换为“xxxx 年 xx 月 xx 日”格式，代码如下：

```
<?php
 include("conn.php");                        //包含数据库连接文件
 $rs=new com("adodb.recordset");
 $rs->open("select bookname,syassort,price,number,format(analysetime,'yyyymmdd') as newdate from tb_booksort",$conn,3,1);
 if($rs->eof|| $rs->bof){
 ?>
<tr>
  <td height="25" colspan="8" bgcolor="#FFFFFF"><div align="center">图书名称</div></td>
</tr>
<?php
  } else{
      while(!$rs->eof) {
?>
<tr>
 <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields(bookname); echo iconv("gbk","utf-8",$f->value);?></div></td>
  <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields(price);echo iconv("gbk","utf-8",$f->value);?></div></td>
    <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields(number); echo iconv("gbk","utf-8",$f->value);?></div></td>
    <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields(syassort); echo iconv("gbk","utf-8",$f->value);?></div></td>
    <td height="25" bgcolor="#FFFFFF"><div align="center"><?php $f=$rs->fields(newdate); echo iconv("gbk","utf-8",$f->value);?></div></td>
</tr>
```

```
            <?php
                $rs->movenext;
                }
            }
        ?>
        </table></td>
    </tr>
</table>
<?php
 }
?>
```

技术要点

format()函数用于返回 variant(string)，其中含一个表达式，它是根据格式表达式中的格式化字符串来确定新格式输出的。该函数的语法格式如下：

```
format(expression[,format [,firstday of week [,firstweek] or year]])
```

参数说明：

☑ expression：必选参数，是任何有效表达式。

☑ format：可选参数。是有效的命名表达式或用户自定义的格式表达式。

☑ firstday of week：可选参数。表示一个星期的第一天。

☑ firstweek of year：可选参数。表示一年的第一周。

实例 126　在查询中使用日期函数

（实例位置：配套资源\SL\06\126）

实例说明

本实例将根据员工的出生日期自动的计算员工年龄并显示结果，运行本实例，如图 6.32 所示，单击图中的“查询”按钮即可将所有员工信息显示出来。

在查询中使用日期函数

明日科技

根据生日查询员工年龄 查询

员工编号	姓名	性别	年龄	祖籍
yg001	小赵	男	26	吉林长春
yg002	小王	女	29	辽宁沈阳
yg003	小科	男	27	吉林四平

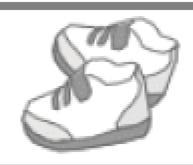
图 6.32　查询员工生日信息

实现过程

具体步骤如下：

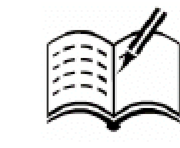
（1）创建 conn.php 文件，实现与数据库的连接，代码如下：

```
<?php
$conn=new com("adodb.connection");                          //连接数据库
```



```
$connstr="provider=microsoft.jet.oledb.4.0;data source=".realpath("./data/db_database06.mdb");
$conn->open($connstr);
?>
```

（2）执行查询，并通过函数 DateDiff()计算所有员工的年龄，显示查询结果，代码如下：



```
<?php
 include("conn.php");                                    //包含数据库连接文件
 $rs=new com("adodb.recordset");
//查询结果
    $rs->open("select ygid,name,sex,address,DateDiff('yyyy',birthday,DATE()) as age from tb_worker",
$conn,3,1);
 if($rs->eof|| $rs->bof) {
 ?>
<tr>
  <td height="25" colspan="8" bgcolor="#FFFFFF"><div align="center"></div></td>
</tr>
<?php
  }else{
      while(!$rs->eof){
?>
<tr>
 <td  height="25"  background="../089/images/head.JPG"><div  align="center"><?php  $f=$rs->
fields("ygid"); echo iconv('gbk','utf-8',$f->value);?></div></td>
  <td  height="25"  background="../089/images/head.JPG"><div  align="center"><?php  $f=$rs->
fields("name"); echo iconv('gbk','utf-8',$f->value);?></div></td>
     <td  height="25"  background="../089/images/head.JPG"><div  align="center"><?php
$f=$rs->fields("sex"); echo iconv('gbk','utf-8',$f->value); ?></div></td>
     <td  height="25" background="../089/images/head.JPG"><div align="center"><?php $f=
$rs->fields("age"); echo iconv('gbk','utf-8',$f->value);?></div></td>
     <td  height="25"  background="../089/images/head.JPG">25<div  align="center"><?php
$f=$rs->fields("address"); echo iconv('gbk','utf-8',$f->value);?></div></td>
</tr>
<?php
     $rs->movenext;
  }
 }
?>
 </table></td>
</tr>
</table>
<?php
 }
?>
```

技术要点

DateDiff()函数返回两个指定日期的时间间隔。该函数的语法格式如下：

```
DateDiff(interval ,date1, date2 [,firstday of week [,firstweek of year]])
```





参数说明：

☑ interval：必选参数，字符串表达式，表示用来计算 date1 和 date2 的时间间隔。

☑ data1,date2：必选参数，用于指定具体日期。

☑ firstday week：可选参数，指定一年的第一周的常数，如果未指定，则以包含 1 月 1 日的星期为第一周。

多学两招：

搜索功能是一个很主要且常见的功能。一个好的搜索功能能给用户使用带来方便。接下来的实例 127 ~ 实例 129 中将介绍一般搜索、高级搜索和常用搜索的实现过程。

实例 127　一般搜索

（实例位置：配套资源\SL\06\127　视频位置：配套资源\SP\06\127）

实例说明

站内搜索有很多种，程序开发人员可以根据站内信息的多少来设置搜索的范围大小。本实例主要介绍一般搜索功能是如何实现的。本实例主要应用 SQL 语言中 select 语句的模糊查询（like）完成站内搜索。运行本实例，在文本框中输入要查询的关键字，然后单击“搜索”按钮，将检索指定条件的记录信息并输出到浏览器，运行结果如图 6.33 所示。



图 6.33　一般搜索运行结果

实现过程

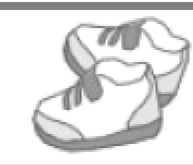
具体步骤如下：

（1）建立与数据源的连接，代码如下：

```
<?php
$link=mysql_connect("localhost","root","111");          //设置数据库连接参数
mysql_select_db("db_database06",$link);                 //选择数据库
mysql_query("set names gb2312");                        //设置编码格式
?>
```

（2）添加 form 表单、文本框，并设置相关属性值，代码如下：

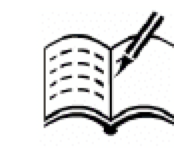
```
<form name="myform" method="post" action="">
  <input name="txt_tj" type="text" id="txt_tj" size="50">
```



```
    <input name="submit" type="submit" id="submit" value="搜索" onClick="return Mycheck();" >
  </form>
```

（3）通过 select 语句中的 like 关键字对输入的关键字进行模糊查询，程序代码如下：

```
<?php
if(isset($_POST["submit"])){                                    //判断用户是否执行提交操作
  $txt_tj=trim($_POST["txt_tj"]);                               //获取去除连续空白的 POST 传递值
  $sql=mysql_query("select * from tb_book where synopsis like '%$txt_tj%' or bookname like
'%$txt_tj%' order by id desc");                                 //执行数据库通配符查询操作
  $info=mysql_fetch_array($sql);                                //遍历结果集
}
if(isset($info)){
do{                                                             //循环输出查询结果
?>
<tr>
     <td height="23"> <font color="#CC0033"> ◎  <?php echo $info[bookname];?>
</font></td>
  </tr>
<tr>
   <td>     <?php echo $info["synopsis"];?></td>
</tr>
 <?php }while($info=mysql_fetch_array($sql));                   //循环输出遍历结果集
 }else{
 ?>
 <tr>
 <td>对不起，您检索的信息不存在！</td>
 </tr>
<?php
}
?>
```



技术要点

本实例在进行数据查询时主要应用 select 语句中的 like 关键字和 or 关键字，进行多条件模糊查询。

like 属于较常用的比较运算符，通过它可以实现模糊查询。它有两种通配符：“%”和下划线“_”。

“%”可以匹配一个或多个字符，而“_”只匹配一个字符。

例如，查找所有第二个字母是“h”的图书，代码如下：

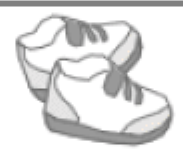
```
select * from tb_mrbook where bookname like('_h%');
```

多学两招：

“p”和“入”都算做一个字符，在这一点上英文字母和中文没有什么区别。

like 运算符用于设置与查询关键字进行模式匹配。例如，要查询 tb_book 表中内容中含有“PHP”的记录，其程序代码如下：

```
select * from tb_book where synopsis like '%PHP%' or bookname like '%PHP%'
```


Note

上面代码中使用的是“%”通配符，select 语句的通配符有以下 4 种。

（1）%（百分号）可以匹配任意个字符。

（2）_()可以匹配任意一个字符。

（3）[ ]()用来限定任何一个单一字符介于赋值的范围或集合中。

（4）IN()关键字用于过滤记录是否为 IN 表达式中任何一个。例如，将 tb_book 表中所有 PHP 或 JSP 的图书信息显示出来的代码如下。

```
SELECT * FROM tb_book WHERE   resume   IN('PHP','JSP')
```

or 关键字用于联合多个条件进行查询，只要满足多个查询条件中的任意一个，这样的记录就会被查询出来。只有那些不满足查询条件中任意一个的记录才会被排除。or 关键字的语法如下：

```
result = expression1 or expression2 [ … and expression n ]
```

参数说明：

☑ result：任意数值变量。

☑ expression1：任意表达式。

☑ expression2：任意表达式。

☑ expression n：任意多个表达式。

实例 128　高级搜索

（实例位置：配套资源\SL\06\128）

实例说明

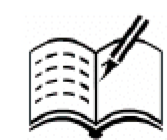
在开发搜索网站时，为了方便用户浏览网站信息，需要添加一些相关信息的搜索功能。本实例就是一个搜索网站，该网站含有大量的数据信息，因此设置高级搜索功能。高级搜索主要通过复合条件查询实现搜索功能。运行本实例，在“请输入要检索的全部名称”文本框和“请输入要检索的部分内容”文本框中输入关键字，在“请选择要搜索的类型”下拉列表框中选择要搜索的类型，然后单击“高级搜索”按钮，对指定条件的记录进行检索并输出结果集到浏览器，运行结果如图 6.34 所示。



图 6.34　高级搜索







实现过程

具体步骤如下：

（1）添加 form 表单、文本框、下拉列表，并设置其相关属性值，关键代码如下：

```
<form name="myform" method="post" action="">
<input name="txt_name" type="text" id="txt_name" size="20">
<input name="txt_content" type="text" id="txt_content" size="20">
<select name="select">
        <option value="ASP">ASP</option>
        <option value="PHP" selected>PHP</option>
        <option value="JSP">JSP</option>
</select>
<input type="submit" name="submit" value="高级搜索" onClick="return Mycheck();">
</form>
```

（2）应用 JavaScript 脚本自定义一个 Mycheck()函数，实现对表单提交信息进行验证，代码如下：

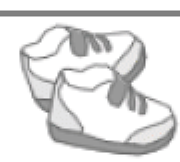
```
<script language="javascript">
function chkinput(form){
  if(form.txt_name.value==""){
                alert("请输入要检索的全部名称!");
                form.txt_name.focus();
                return(false);
          }
          if(form.txt_content.value==""){
                alert("请输入要检索部分的内容!");
                form.txt_content.focus();
                return(false);
          }
          return(true);
}
</script>
```

（3）获取表单中提交的查询关键字，应用 like 比较运算符和 and 关键字完成多条件的模糊查询。其关键代码如下：

```
<?php
if(isset($_POST["submit"])){        //判断提交按钮是否设置
  $txt_name=trim($_POST["txt_name"]);        //获取查询关键字
  $txt_content=trim($_POST["txt_content"]);
  $type=$_POST["select"];
  $sql=mysql_query("select * from tb_mrbook where bookname like '%$txt_name%' and  synopsis
like '%$txt_content%' and resume like '%$type%'order by id desc");//执行多条件的模糊查询
  $info=mysql_fetch_array($sql);    //获取查询结果集
}
```

技术要点

本实例主要应用复合条件查询，完成高级搜索功能。应用 and 关键字对两个表达式进



行逻辑与运算，来实现高级搜索。

and 关键字用于联合多个条件进行查询。使用 and 关键字时，只有同时满足所有查询条件的记录会被查询出来。如果不满足这些查询条件中的任意一个，则这样的记录将被排除。and 关键字的语法如下：

```
result = expression1 and expression2 [ … and expression n ]
```

参数说明：

- ☑ result：任意数值变量。
- ☑ expression1：任意表达式。
- ☑ expression2：任意表达式。
- ☑ expression n：任意多个表达式。

另外，本实例还支持数据库数据检索描红技术，即对查询到的数据标题进行描红操作。其实现原理是：当用户单击“搜索”按钮对数据搜索时，将返回的数据添加标签<font>，并通过<font>标签设置返回数据的颜色为红色。其关键代码如下：

```
<font color="#CC0033">◎ <?php echo $info["bookname"];?></font>
```

实例 129　程序员搜索引擎

（实例位置：配套资源\SL\06\129）

实例说明

当提到“搜索引擎”这个名词时，自然会想到百度、谷歌等一些非常著名的搜索引擎网站。本实例中将介绍一个简单搜索引擎网站的制作。其常用搜索主要包括人才网站搜索、IT 企业故事搜索、IT 企业风云人物简介搜索、常用术语搜索、开发语言简介搜索、编程网站搜索、图书资源搜索、创业故事搜索。其中开发语言搜索的运行结果如图 6.35 所示。



图 6.35　程序员搜索引擎





实现过程

具体步骤如下：

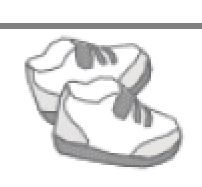
（1）创建 conn.php 文件，设置数据库连接参数，完成与指定数据库的连接操作。其代码如下：

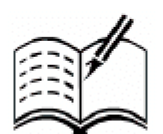
```
<?php
$link=mysql_connect("localhost","root","111");          //设置数据库连接参数
mysql_select_db("db_database06",$link);                 //选择数据库
mysql_query("set names gb2312");                        //设置数据库编码格式
?>
```

（2）创建 index.php 文件，应用 mysql_query()函数执行 SQL 查询语句，应用 do…while 循环输出 mysql_fetch_array()函数获取的查询结果集。并且创建自定义函数 chinesesubstr()对数据的内容进行截取输出。其关键代码如下：

```
<table width="555" border="0" cellpadding="0" cellspacing="0">
<?php
$sql=mysql_query("select distinct resume,info from tb_books order by id desc");//执行查询语句
$info=mysql_fetch_array($sql);                          //获取结果集
if($info){
 do{                                                    //循环输出结果集
?>
 <tr>
     <td width="555" height="23">
         <font color="#CC0033">◎  
             <a href="info.php?resume=<?php echo $info["resume"];?>"><?php echo $info
["resume"];?></a>
          </font>
         </td>
  </tr>
    <tr>
     <td><?php echo chinesesubstr($info["info"],0,300);if(strlen($info["info"])>300){echo "...";}?>
</td>
     </tr>
<?php
 }while($info=mysql_fetch_array($sql));
}else{
?>
 <tr>
     <td>对不起，您检索的信息不存在！</td>
     </tr>
<?php
}
?>
</table>
```

（3）单击 index.php 文件中设置的类别名称超链接，将跳转到 info.php 页，根据超链接传递的类别名称执行查询语句，查询出符合条件的图书信息。







技术要点

本实例意在让读者了解简单的搜索引擎，同时让读者了解PHP数据库编程的基本步骤以及基本的函数应用。

（1）通过mysql_connect()函数连接MySQL服务器。

（2）通过mysql_select_db()函数选择指定数据库。

（3）通过mysql_query()函数设置数据库编码格式。

（4）通过mysql_query()函数执行SQL语句。

（5）通过mysql_fetch_array()函数获取查询结果集。

（6）通过mysql_close()函数关闭数据库。

在本实例中还引入了一个自定义函数，对中文字符串进行截取操作。chinesesubstr()函数的语法如下：

```
function chinesesubstr($str,$start,$len){   //$str指的是字符串，$start指的是字符串的起始位置，$len指的是长度
    $tmpstr="";
    $strlen=$start+$len;//用$strlen存储字符串的总长度（从字符串的起始位置到字符串的总长度）
    for($i=0;$i<$strlen;$i++){              //通过for循环语句，循环读取字符串
        if(ord(substr($str,$i,1))>0xa0){//如果字符串中首个字节的ASCII序数值大于0xa0，则表示为汉字
            $tmpstr.=substr($str,$i,2);     //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
            $i++;                           //变量自加1
        }else{                              //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
            $tmpstr.=substr($str,$i,1);
        }
    }
    return $tmpstr;                         //输出字符串
}
```

应用此函数的目的是避免在截取中文字符串时出现乱码。其原理是：对截取字符串的ASCII值进行判断，如果值大于“0xa0”，说明是中文，那么以2字节为截取长度对原字符串执行截取操作，否则应用substr()函数正常截取字符串。

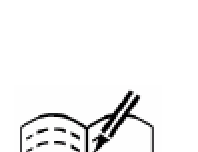


字符串高级处理

本章读者可以学到如下实例：

- 实例 130　统计帖子标题的长度
- 实例 131　购物车中数据的读取
- 实例 132　查询关键字描红
- 实例 133　统计查询关键字的出现次数
- 实例 134　去除帖子标题的首尾空格
- 实例 135　验证电话号码的格式是否正确
- 实例 136　验证 E-mail 地址格式是否正确
- 实例 137　超长文本的分页输出
- 实例 138　统一上传文件名称的大小写
- 实例 139　货币数据的格式化输出
- 实例 140　统一英文注册用户首字母大小写
- 实例 141　解决用 substr()函数对中文字符串截取时乱码
- 实例 142　将元素中指定位置的索引替换
- 实例 143　统计数组元素的个数
- 实例 144　去除数组中的重复元素
- 实例 145　判断字符串中是否存在指定子串
- 实例 146　合并数组
- 实例 147　拆分数组
- 实例 148　将时间和日期转换为时间戳
- 实例 149　网页闹钟





实例130　统计帖子标题的长度

（实例位置：配套资源\SL\07\130）

实例说明

统计字符串长度一般是为了给其他函数的应用做好铺垫。本实例通过 strlen()函数获取帖子标题的长度。注意，由于页面采用 UTF-8 编码格式，所以一个中文字符串长度是 3 个字节；如果页面采用 GB2312 编码格式，那么一个中文字符串长度是 2 个字节。其运行结果如图 7.1 所示。

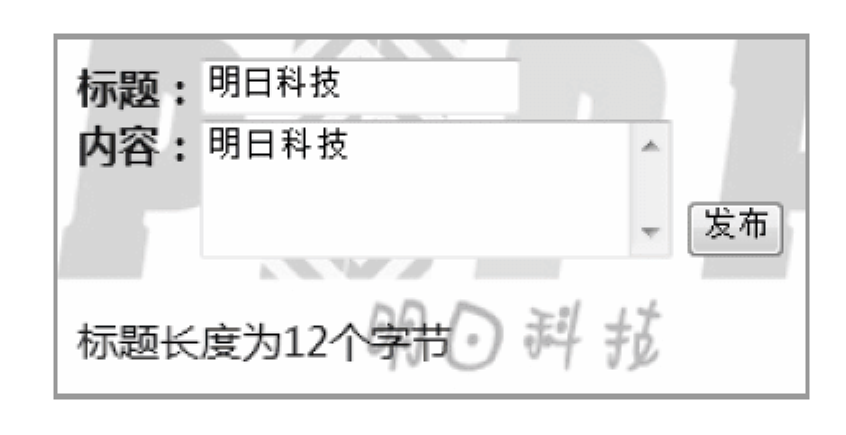


图 7.1　统计帖子标题的长度

实现过程

创建名称为“index.php”的 PHP 脚本文件，当用户单击“发布”按钮时，利用 strlen()函数获取通过 POST 方法接收到的数据长度，其代码如下：

```
<?php
 if(!empty($_POST["sub"])){                                     //通过 POST 方式传参
        echo "标题长度为".strlen($_POST["text"])."个字节";       //获取字符串长度
 }
?>
```

技术要点

strlen()函数用于获取字符串的长度。其语法如下：

```
int strlen ( string str);
```

参数 str：指定字符串。

多学两招：

用 strlen()函数获取字符串长度时，如果字符串中存在空格，则空格也会被计算在内。

实例131　购物车中数据的读取

（实例位置：配套资源\SL\07\131）

实例说明

SESSION 购物车的设计原理：当用户登录网站时，系统将为每个用户分配两个 SESSION 变量$producelist 和$quatity，分别用来存储用户放入购物车中的商品 ID 和商品数量，将@作为分隔符，实现将多个 ID 值同时保存在一个$producelist 变量中。例如，用户分别将 ID 为 1、2、3 的商品放入购物车中，这时 SESSION 变量$producelist 的值应该为



“1@2@3@”，这就是 SESSION 购物车的设计原理。其运行结果如图 7.2 所示。



图 7.2　购物车中商品展示

实现过程

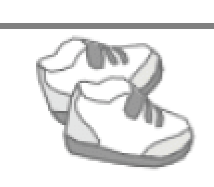
创建名称为“gouwu.php”的 PHP 文件，应用 explode()函数读取 SESSION 变量中存储的字符串，并返回数组元素；然后，应用 for 循环语句读取数组中的元素值，将获取的元素值作为条件，执行查询语句，从数据库中读取数据；最后，完成商品金额的计算，并输出购买商品信息。

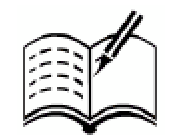
关键代码如下：

```
<?php
$total=0;
$array=explode("@",$_SESSION[" producelist"]);     //读取 SESSION 中存储的商品 ID，返回数组
$arrayquatity=explode("@",$_SESSION["quatity"]); //读取 SESSION 中存储的商品数量，返回数组
for($i=0;$i<count($array)-1;$i++){                 //for 循环获取商品的数量和 ID
  $id=$array[$i];                                  //获取商品 ID
  $num=$arrayquatity[$i];                          //获取商品数量
  if($id!=""){                                     //判断是否存在指定的商品
        $sql=mysql_query("select * from tb_shangpin where id='".$id."'",$conn);
        $info=mysql_fetch_array($sql);             //获取查询结果
        $total1=$num*$info["huiyuanjia"];          //计算商品金额
        $total+=$total1;                           //计算总的金额
        $_SESSION["total"]=$total;                 //为 SESSION 变量赋值
?>
```

技术要点

SESSION 购物车的设计原理：当用户登录网站时，系统将为每个用户分配两个 SESSION 变量，它们分别是$producelist 和$quatity，二者分别用来存储用户放入购物车中的商品 ID 和商品数量，将@作为分隔符，实现将多个 ID 值同时保存在一个$producelist 变量中。例如，用户分别将 ID 为 1、2、3 的商品放入购物车中，这时 SESSION 变量$producelist 的值应该为“1@2@3@”。另外，如果需要设置购买某种商品数量时，应该将带有商品 ID 的变量 $SESSION ["producelist"]与商品数量$SESSION ["quatity"]中的位置一一对应。如打算更改 ID 为 2 的商品数量 3 个，只需将$SESSION ["producelist"]中“1@2@3@”ID 为 2 所对应$SESSION ["quatity "]的位置改为 3，即“1@3@1@”，来实现购物车中某种商品数量的更改。





Note

多学两招：

SESSION和Cookie一样存在其生命周期，如用户需要更改SESSION存在的时间长度，可以通过应用session_set_cookie_params()函数来延长SESSION的生命周期。

实例 132　查询关键字描红

（实例位置：配套资源\SL\07\132　视频位置：配套资源\SP\07\132）

实例说明

查询关键字描红在百度、谷歌等搜索引擎中是非常实用的，通过它可以凸显出要查找的内容。运行结果如图 7.3 所示。



图 7.3　查询关键字描红

实现过程

应用str_ireplace()函数，将指定输入的内容用带有CSS属性的font标签的原内容替换，来实现字符串关键字的描红。其关键代码如下：

```
<?php
$string=str_ireplace($txt_tj,"<font 
color='#FF0000'><strong>$txt_tj</strong></font>",$info[bookname]);
echo $string;
?>
<?php
$strings=$_POST["txt_tj"];                          //作为加入CSS标签的替换内容
$string=str_ireplace($_POST["txt_tj"],"<font color='#FF0000'><strong>$strings</strong></font>",$info
[synopsis]);
echo $string;                                       //输出描红的字符串
?>
```

技术要点

查询关键字描红关键就是应用 str_ireplace()函数对提交的关键字进行替换，即首先为



关键字进行包装，然后应用 str_ireplace()函数用包装后的关键字替换掉原来未包装的关键字。其包装的手法就是将关键字的字体加粗，并设置为红色。



Note

多学两招：

用 strlen()函数获取字符串长度时，如果字符串中存在空格，则空格也会被计算在内。

实例 133　统计查询关键字的出现次数

（实例位置：配套资源\SL\07\133）

实例说明

在站内搜索中，我们往往需要列出符合条件的关键字有多少个。本实例通过字符串函数 substr_count()统计查询关键字的出现次数，运行结果如图 7.4 所示。

被搜索的文本：PHP作为全球最普及、应用最广泛的互联网开发语言之一，从1994年诞生至今已被2000多万个动态网站采用，全球知名互联网公司Google、Yahoo、eBay和中国知名网站新浪、百度、阿里巴巴等均采用PHP技术！

关键字：PHP 搜索

PHP作为全球最普及、应用最广泛的互联网开发语言之一，从1994年诞生至今已被2000多万个动态网站采用，全球知名互联网公司Google、Yahoo、eBay和中国知名网站新浪、百度、阿里巴巴等均采用**PHP**技术！

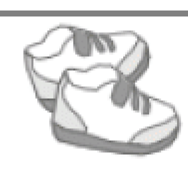
关键字出现2次。

图 7.4　统计查询关键字的出现次数

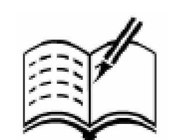
实现过程

创建脚本文件 index.php。首先，定义字符串变量，编写 form 表单。当单击“搜索”按钮时，利用 substr_count()函数输出指定字符串子串的出现次数。其代码如下：

```
<?php
echo "<a style='font-size:20px; color:#000000'>被搜索的文本：</a>";        //定义字符串输出
$str = "PHP 作为全球最普及、应用最广泛的互联网开发语言之一，从 1994 年诞生至今已被 2000 多万个动态网站采用，全球知名互联网公司 Google、Yahoo、eBay 和中国知名网站新浪、百度、阿里巴巴等均采用 PHP 技术！";
cho $str."<br>";
?>
<form action ="" method="post">                                              //表单
  关键字：<input type="text" name="text"><input type="submit" name="sub" value ="搜索">
</form>
</body>
</html>
<?php
if(!empty($_POST["sub"])){                                                  //通过 POST 方式传递参数
  $a = "<b style='color:red;font-size:18px;'>".$_POST["text"]."</b>";  //替换后的字符串
  echo str_replace("$_POST["text"]",$a,$str)."<br>";                    //替换字符串
  //统计关键字出现次数
```



```
    echo "关键字出现<b style='color:red;font-size:22px;'>".substr_count($str,$_POST["text"])."</b>次。";
  }
  ?>
```



技术要点

本实例通过 substr_count()函数检索字符串子串的出现次数。substr_count()函数的语法如下：

```
int substr_count ( string haystack, string needle )
```

参数说明：

☑ string haystack：源字符串。

☑ string needle：字符串子串。

多学两招：

substr_count()是一种类似全文检索的函数。它的原理是检索查找匹配的字符串，并返回匹配的次数。

实例 134　去除帖子标题的首尾空格

（实例位置：配套资源\SL\07\134）

实例说明

用户在编写代码时，也许会在不经意间多了一个或多个空格，以致程序无论怎么调试都无法正常运行。本实例通过 trim()函数实现去除帖子标题的首尾空格，运行结果如图 7.5 所示。

标题： 明日科技 提交
未去掉空格前字符长度为18个字节
去掉空格后字符长度为12个字节

图 7.5　去除帖子标题的首尾空格

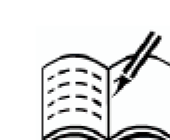
注意，由于页面采用 UTF-8 编码格式，所以一个中文字符串的长度是 3 个字节，而一个空格的长度是 2 个字节，所以本实例去掉空格前的长度为 18 个字节，去掉空格后的长度为 12 个字节。读者可以自行创建一个 GB2312 编码格式的网页，然后实现此功能，比较一下它们之间有什么不同。

实现过程

创建 PHP 脚本文件。当用户单击“提交”按钮时，利用 strlen()函数获取函数的长度，然后应用 trim()函数和 strlen()函数的联合操作，实现输出在去除首尾空格的前后字符串的长度。其代码如下：

```
<?php
if(!empty$_POST["sub"])){                                                //通过 POST 方式传递参数
  echo "未去掉空格前字符长度为".strlen($_POST["text"])."个字节<br>";        //输出结果
  echo "去掉空格后字符长度为".strlen(trim($_POST["text"]))."个字节";        //输出结果
}
?>
```





技术要点

在 PHP 中去除字符串的首尾空格，可以使用函数 trim()实现。Trim()函数的语法如下：

```
string trim(string str);
```

参数 string str 为需要过滤空格的字符串。

多学两招：

PHP 还提供了单纯去除字符串左边空格的函数 ltrim ()和单纯去除右边空格的函数 rtrim()。

实例 135　验证电话号码的格式是否正确

（实例位置：配套资源\SL\07\135　视频位置：配套资源\SP\07\135）

实例说明

表单注册时往往要求用户书写座机电话号码，而座机电话号码是由 12 位或 11 位数字组成的，所以一定要对电话号码的位数和格式进行限制。本实例通过正则表达式和正则表达式函数 preg_match()实现对电话号码格式的验证，运行结果如图 7.6 所示。

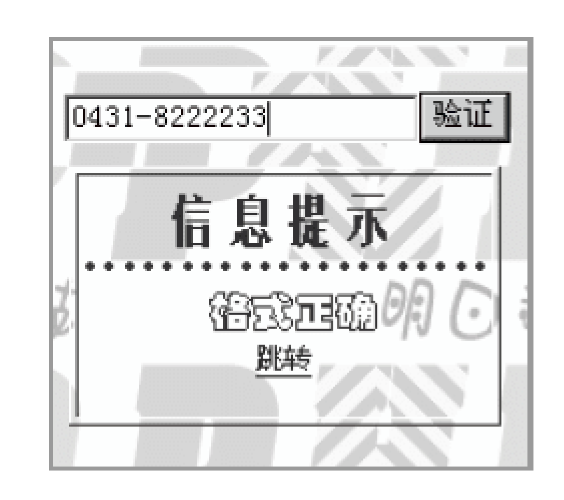


图 7.6　验证电话号码的格式是否正确

实现过程

创建 index.php 文件。创建表单，并在页面的表单中设置用户文本框和“提交”按钮，当用户单击“提交”按钮时，通过正则表达式函数 preg_match()对用户在文本框输入的电话号码进行验证，并输出相关提示，其代码如下：

```
<?php
if(!empty($_POST['sub'])){                                        //判断是否存在传递参数
  if(preg_match("/(\d{3}-)(\d{8})$|(\d{4}-)(\d{7})$/",$_POST['text'])){//正则表达式验证
        include("inc.php");                                       //包含信息提示页
        show_error("信息提示","格式正确","index.php");       //调用自定义函数输出提示信息
  }else{
        include("inc.php");
        show_error("信息提示","格式错误","index.php");       //调用自定义函数输出提示信息
            }
        }
?>
```

脚下留神：

这里给出的实例仅用来判断区号为 4 位的电话号码。当电话号码的区号为 3 位时，如“010-12345678”，则不能通过正则表达式的验证。



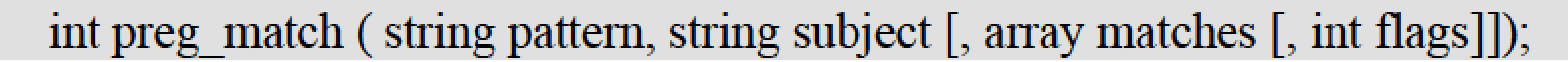
技术要点

（1）preg_match()：进行正则表达式匹配。

```
int preg_match ( string pattern, string subject [, array matches [, int flags]]);
```

（2）正则表达式：固定总位数为 11 位或 12 位的座机电话。

```
/(\d{3}-)(\d{8})$|(\d{4}-)(\d{7})$/;
```



多学两招：

正则表达式“/(\d{3}-)(\d{8})$|(\d{4}-)(\d{7})$/”是根据新式电话号码设置的，老式的座机电话是 10 位数字组成的。

实例 136　验证 E-mail 地址格式是否正确

（实例位置：配套资源\SL\07\136　视频位置：配套资源\SP\07\136）

实例说明

互联网发展到今天，几乎所有的 Web 爱好者都有自己的 E-mail 地址，无论你申请的是 126 邮箱还是 163 邮箱，E-mail 地址的格式都是固定的。本实例通过 preg_match()正则匹配函数和正则表达式验证 E-mail 地址格式是否正确，运行结果如图 7.7 所示。



图 7.7　验证 E-mail 地址格式是否正确

实现过程

本实例的实现过程与实例 135 大同小异。首先，创建 index.php 文件并设置带有文本框的表单，当用户单击“验证”按钮时，处理文件会通过正则表达式函数验证文本框输入的信息并输出相关提示。其代码如下：

```
if($_POST['sub']){
  require_once("inc.php");                                    //包含信息提示页
  if(preg_match("/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/",trim($_POST['text']))){//正则表达式验证
        show_error("信息提示","格式正确","index.php");              //输出提示
  }else{
        show_error("信息提示","格式不正确","index.php");            //输出提示
  }
}
?>
```

技术要点

（1）preg_match()：进行正则表达式匹配。用于匹配指定字符串，一般会与 if 条件语







句协同使用。其语法如下：

```
int preg_match ( string pattern, string subject [, array matches] )
```

函数功能：在字符串 subject 中匹配表达式 pattern，函数返回匹配的次数。如果有数组 matches，那么每次匹配的结果将被存储到数组 matches 中。该函数在匹配成功后就停止继续查找，其返回值是 0 或 1。

（2）正则表达式：对 E-mail 地址进行验证。

```
/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/;
```

多学两招：

PHP 支持两种正则表达式函数库：一种是正则表达式函数库（POSIX 扩展），另外一种是正则表达式函数库（Perl 兼容）。在性能上，与 Perl 兼容相比，正则表达式速度更快一些。

实例 137　超长文本的分页输出

（实例位置：配套资源\SL\07\137　视频位置：配套资源\SP\07\137）

实例说明

在遍历文件的内容时，由于文件内容很多，最理想的方法就是分页输出文本中的内容。本例将介绍如何将超长文本分页输出，其运行效果如图 7.8 所示。



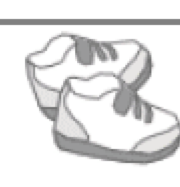
图 7.8　超长文本的分页输出

实现过程

具体步骤如下：

（1）创建 function.php 文件，编写自定义函数 msubstr()，完成对文本文件的截取操作。

（2）创建 index.php 文件，首先通过文件系统函数 file_get_contents()读取整个文件的内容，然后调用自定义函数和字符串函数；完成对文件的截取操作，并实现截取后内容的



分页输出。其关键代码如下：

```
<?php
if (isset($_GET['page'])) {                          //定义初始变量的值
  $page=$_GET['page'];
}else{
  $page=1;                                           //变量值为 1
}
include("function.php");                             //调用自定义函数文件
?>
  <?php
//读取超长文本中的数据，实现超长文本中数据的分页显示
if($page){
  $counter=file_get_contents("file/file.txt");       //读取文本文件
      $length=strlen(unhtml($counter));//获取文本文件的长度，并通过自定义函数去除特殊字符和
空格
      $page_count=ceil($length/850);                 //对超长文本进行分页，每页显示 850 个字符
      $c=msubstr($counter,0,($page-1)*850);          //计算上一页的字节
      $c1=msubstr($counter,0,$page*850);             //计算下一页的字节
       echo substr($c1,strlen($c),strlen($c1)-strlen($c));    //获取当前的字节
}
?></td>
  <td rowspan="2"> </td>
</tr>
<tr>
  <td><span class="STYLE3">页次：<?php echo $page;?> / <?php echo $page_count;?>  页  </span>
        分页：
  <?php
   if($page!=1){
          echo   "<a href=index.php?page=1>首页</a> ";
          echo "<a href=index.php?page=".($page-1).">上一页</a> ";
 }
  if($page<$page_count){
       echo "<a href=index.php?page=".($page+1).">下一页</a> ";
          echo   "<a href=index.php?page=".$page_count.">尾页</a>";
       }
?>
```

技术要点

完成超长文本的分页输出需要三方面的技术：第一方面，自定义函数。通过自定义函数读取文本文件，可以避免中文字符串出现乱码；第二方面，字符串函数。需要通过 strlen()函数计算字符串的长度，通过 substr()函数对字符串进行截取；第三方面，文件系统函数。通过 file_get_contents()函数读取文本文件中的数据。

自定义函数 msubstr()的语法如下：

```
function msubstr($str,$start,$len){ //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的
是长度
  $tmpstr="";
  $strlen=$start+$len;//用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
```



```
        for($i=0;$i<$strlen;$i++){              //通过 for 循环语句，循环读取字符串
            if(ord(substr($str,$i,1))>0xa0){//如果字符串中首个字节的 ASCII 序数值大于 0xa0，则表示
为汉字
                $tmpstr.=substr($str,$i,2);     //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
                $i++;                           //变量自加 1
            }else{                              //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
                    $tmpstr.=substr($str,$i,1);
            }
        }
        return $tmpstr;                         //输出字符串
    }
```

多学两招：

如果文字信息的数量过多，最好使用分页来显示。在每一页中显示指定数量的文字信息，如果超过指定数量就在下一页中显示。

实例 138　统一上传文件名称的大小写

（实例位置：配套资源\SL\07\138）

实例说明

统一上传文件名称的大小写是十分必要的。例如，上传图片文件 img.jpg 和 Img.jpg，如果文件名称未经大小写统一，上传时可能就不会出现覆盖提示。本实例通过字符串函数 strtoupper()、strtolower() 实现统一上传文件名称的大小写，运行结果如图 7.9 所示。

关键字：RTX Files\P1020094w.jpg 浏览… 上传

文件名称自动转换为大写：P1020094W.JPG
文件名称自动转换为小写：p1020094w.jpg

图 7.9　统一上传文件名称的大小写

实现过程

具体步骤如下：

（1）创建 PHP 脚本文件。当单击“上传”按钮时，分别利用转换小写和大写的函数 strtolower()、strtoupper()将接收到的文本框的信息进行对应的大小写转换，并输出转换结果。其代码如下：

```
    <?php
        if($_POST[sub]){                                        //通过 POST 方式传参
            $a = strtoupper($_POST[text]);                      //转换为大写
            echo "文件名称自动转换为大写：".$a."<br>";           //输出
            $b = strtolower($_POST[text]);                      //转换为小写
            echo "文件名称自动转换为小写：".$b;                  //输出
        }
    ?>
```

（2）将该文件存储于\MR\02\091\文件夹下，并命名为 index.php。运行结果如图 7.9



所示。

技术要点

实现字符串的大小写转换，主要通过 strtolower()和 strtoupper()函数。

（1）strtolower()：转换为小写。

```
string strtolower ( string str )
```

（2）strtoupper()：转换为大写。

```
string strtoupper ( string string )
```

参数 string 为需要转换的字符串。



实例 139　货币数据的格式化输出

（实例位置：配套资源\SL\07\139）

实例说明

货币数据不同于整型数据，货币数据是存在一定格式的。本实例通过函数 number_format()实现货币数据的格式化输出。运行结果如图 7.10 所示。



图 7.10　货币数据的格式化输出

实现过程

实现此格式化输出的过程相当简单。创建名称为“index.php”的 PHP 脚本文件，当用户输入需要格式化的数字并单击“格式化”按钮时，程序自动将文本框中的信息格式化。代码如下：

```
<?php
if(!empty($_POST["sub"])){                    //通过 POST 方式传递参数
  echo number_format($_POST["text"],2);       //格式化数字
}
?>
```

脚下留神：

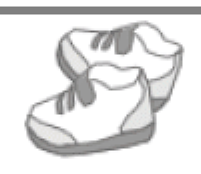
格式化函数 number_format()存在两种结构，这里只应用保留小数点后位数的结构。另一种结构和参数请参看技术要点中的内容。

技术要点

货币数据的格式化输出通过函数 number_format()实现。其语法说明如下：

```
string number_format ( float number , int decimals)
string number_format ( float number, int decimals, string dec_point, string thousands_sep)
```

number_format()函数返回参数 number 格式化后的字符串，该函数可以有 1 个、2 个或





是 4 个参数，但不能是 3 个参数。如果只有 1 个参数 number，number 格式化后会舍去小数点后的值，且每隔 3 位（千位）就会以逗号（,）来隔开；如果有 2 个参数，number 格式化后会到小数点第 decimals 位，且每隔 3 位就会以逗号来隔开；如果有 4 个参数，number 格式化后会到小数点第 decimals 位，dec_point 用来替代小数点（.），thousands_sep 用来替代每 3 位隔开的逗号（,）。

多学两招：

开发电子商务类网站时，函数 number_format()出现的频率是百分之百的，所以掌握此函数是十分必要的。

实例 140　统一英文注册用户首字母大小写

（实例位置：配套资源\SL\07\140）

实例说明

网站的国际化已经不是一个炙手可热的话题，但是由于地区差异，在使用英文填写注册信息时首字母大小写得不到统一。本实例利用函数 ucfirst()实现英文注册用户首字母统一为大写，运行结果如图 7.11 所示。



图 7.11　统一英文注册用户首字母的大小写

实现过程

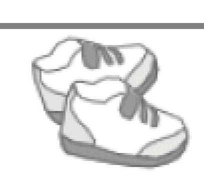
创建 PHP 脚本文件。当用户单击“提交”按钮时，首先利用 POST 方法获取从文本框传递进来的数据。然后将获取的数据经 ucfirst 函数转换为首字母大写，最后输出经转换后的数据。具体代码如下：

```
<?php
If(!empty($_POST["sub"])){                    //通过 POST 方式传参
  $a = $_POST["text"];                        //接收参数
  echo "首字母统一为大写:";
  echo ucfirst($a);                           //转换单词首字母
}
?>
```

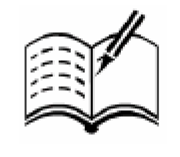
技术要点

在 PHP 开发中定义首字母大写的方法有很多种，但是所有的方法都没有 PHP 预定义函数 ucfirst 简单。ucfirst 的语法如下：

```
string ucfirst ( string str );
```



参数 string str 为定义字符串。



Note

多学两招：

要实现字符串首字母大小写转换还可以通过转码来完成。首先利用函数 substr()提取出字符串的首字母，然后利用函数 ord()获取该字母的 ASCII 码，再根据大小写字母之间相差 32 来实现大小写字母之间的 ASCII 码转换，最后利用函数 chr()将转变后的 ASCII 码转变为字母即可。

实例 141　解决用 substr()函数对中文字符串截取时乱码

（实例位置：配套资源\SL\07\141）

实例说明

substr()函数是按字节进行截取字符串的。在截取中文字符串时，由于一个汉字由两个字符组成，如果只截取 1 个字符就会出现乱码。本实例使用自定义函数解决对中文字符串截取时的乱码问题，运行结果如图 7.12 所示。

吉林省明日科技有限公司是一家以计算机软件技术为核心的高科技型企业

从第 1 字开始截取，截取 11 个字。 截取

吉林省明日科技有限公司

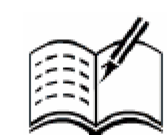
图 7.12　解决用 substr()函数对中文字符串截取时的乱码问题

实现过程

创建 index.php 文件，定义 msubstr()函数，对中文字符串进行截取。创建 form 表单，提交字符串截取的开始位置和长度，然后调用 msubstr()函数对字符串进行截取，并输出截取结果。具体代码如下：

```
<?php
function msubstr($str, $start, $len) { //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
    if($len%2==0){                    //判断输入的数字是否为偶数
        $len=$len*3;                  //根据输入的数字获取汉字所占位数
    }else{
        $len=($len+1)*3-2;            //根据输入的数字获取汉字所占位数
    }
    $strlen = $start + $len; //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
    for($i = 0; $i < $strlen; $i ++) {          //通过 for 循环语句,循环读取字符串
        if (ord ( substr ( $str, $i, 1 ) ) > 0xa0) {  //如果字符串中首个字节的ASCII序数值大于0xa0，则表示为汉字
            $tmpstr .= substr ( $str, $i, 2 );//每次取出两位字符赋给变量$tmpstr，即等于一个汉字
            $i ++; //变量自加 1
        } else {                      //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
```





```
                $tmpstr .= substr ( $str, $i, 1 );
            }
        }
        return $tmpstr;                    //输出字符串
    }
    $string="吉林省明日科技有限公司是一家以计算机软件技术为核心的高科技型企业";
    $str=msubstr($string,$_POST[te],$_POST[tx]);
    echo $str;
    ?>
```

脚下留神：

本实例以汉字为例，在应用自定义函数时，如果不将传入截取字符串的值以原数值 3 倍给出，可能会导致获取的汉字字符数量为原来的一半，故这里必须将传入值以原数 3 倍作为参数给出。

技术要点

自定义函数 msubstr()的语法如下：

```
    function msubstr($str, $start, $len) { //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
        $strlen = $start + $len; //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
        for($i = 0; $i < $strlen; $i ++) {     //通过 for 循环语句，循环读取字符串
            if (ord ( substr ( $str, $i, 1 ) ) > 0xa0) {   //如果字符串中首个字节的ASCII序数值大于0xa0，则表示为汉字
                $tmpstr .= substr ( $str, $i, 2 ); //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
                $i ++; //变量自加 1
            } else {                       //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
                $tmpstr .= substr ( $str, $i, 1 );
            }
        }
        return $tmpstr;                    //输出字符串
    }
```

参数$str 是指定被截取的字符串；参数$strart 是截取的开始位置；参数$len 是截取的长度；返回值$tmpstr 是截取后的字符串。

多学两招：

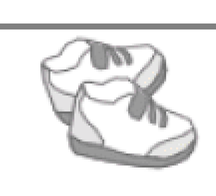
使用 msubstr()截取中文最好先定义算法，以免截取半个中文字而出现乱码。

实例 142　将元素中指定位置的索引替换

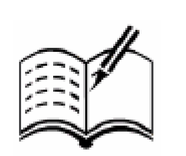
（实例位置：配套资源\SL\07\142）

实例说明

创建数组时，有时会由于疏忽将元素内容书写错误，更改错误的元素内容可以通过数



组键值来完成。本实例通过数组下标（键值）将数组中指定索引位置的元素替换，运行结果如图 7.13 所示。



Note

原数组元素：Array ([0] => 1 [1] => 精细 [2] => asd [3] => asd123)
指定索引元素替换后的数组：Array ([0] => 1 [1] => 精细 [2] => 经济 [3] => asd123)
键值1的元素发生更改

图 7.13　将数组中指定索引位置的元素替换

实现过程

创建 index.php 文件，定义数组变量并通过数组下标输出元素。其代码如下：

```
<?php
echo "原数组元素：";
$a = array('1','精细','asd','asd123');          //定义数组
print_r($a);                                   //打印数组
echo "<br>指定索引元素替换后的数组：";
$a[2] = '经济';                                 //根据数组下标更改元素
print_r($a);                                   //打印更改后的数组
echo "<br>键值 2 的元素发生更改";
?>
```

技术要点

PHP 数字索引由数字组成，下标从 0 开始，数值索引一般表示数组元素在数组中的位置，数字索引数组默认索引值从数字 0 开始，不需要特别指定，PHP 会自动为索引数组的键名赋一个整数值，然后从这个值开始自动增量。当然，也可以指定从某个位置开始保存数据。

多学两招：

当使用数组名称[键值]这样的格式时，要注意数组键值是从 0 开始的，所以本实例中更改键值为 2 的元素其实是数组中的第三个元素。

实例 143　统计数组元素的个数

（实例位置：配套资源\SL\07\143）

实例说明

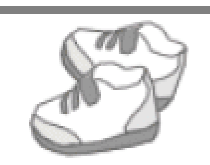
数组下标默认是以 0 为开始键值的整型数据。如果我们想取得数组元素的个数，就要使用 count()函数。本实例通过 count()函数统计数组元素的个数，运行结果如图 7.14 所示。



图 7.14　统计数组元素的个数

实现过程

创建 index.php 文件。首先定义数组变量并为此变量赋值，然后利用 count()函数输出数组元素的个数。其代码如下：



```
<?php
$a = array('1','精细','asd','asd123');                    //声明数组
echo "数组中存在元素".count($a)."个";                      //计算数组元素个数
?>
```

技术要点

在 PHP 中，应用 count()函数可以对数组中的元素的个数进行统计。其语法如下：

```
int count ( mixed array [, int mode])
```

参数说明：

☑ array：必选参数，指定被统计的数组。

☑ mode：可选参数，COUNT_RECURSIVE（或 1），如选中此参数，本函数将递归地对数组计数，对计算多维数组的所有单元尤其有用，此参数的默认值为 0。

多学两招：

如果 count()函数的操作对象是“NULL”，那么返回结果为 0。count()函数对没有初始化的变量返回 0，但对于空的数组也会返回 0。如果要判断变量是否初始化，则可以应用 isset()函数。count()函数不能识别无限递归。

实例 144　去除数组中的重复元素

（实例位置：配套资源\SL\07\144）

实例说明

在数组中，键值是唯一的，但是元素的值是可以重复的。想要删除数组元素中重复的值，可以使用函数 array_unique()。本实例是通过数组函数 array_unique()去除数组中的重复元素值，运行结果如图 7.15 所示。

```
原数组元素为：Array ( [0] => a [1] => b [2] => a [3] => d [4] => a [5] => f [6]
=> a [7] => h [8] => a )
去除重复项后的数组为：Array ( [0] => a [1] => b [3] => d [5] => f [7] => h )
```

图 7.15　去除数组中的重复元素

实现过程

创建 index.php 文件。首先创建数组变量$a 并为$a 赋值，然后利用 array_unique()去除数组中的重复元素值，最后打印数组。其代码如下：

```
<?php
$a = array('a','b','a','d','a','f','a','h','a');          //定义数组
echo "原数组元素为：";
print_r($a);                                              //打印数组
echo "<br>去除重复项后的数组为：";
print_r(array_unique($a));                                //输出去除重复项的数组
?>
```





技术要点

在 PHP 中，可以通过 array_unique()函数去除数组中的重复元素。其语法如下：

```
array array_unique ( array array);
```

参数 array 为指定参数的数组，其返回值为一个没有重复元素的新数组。

多学两招：

用 strlen()函数获取字符串长度时，如果字符串中存在空格，则空格也会被计算在内。

实例 145　判断字符串中是否存在指定子串

（实例位置：配套资源\SL\07\145）

实例说明

判断一个字符串中是否含有另一个字符串方法有很多，例如，正则表达式等。本实例通过 stristr()函数判断字符串中是否存在指定子串，运行结果如图 7.16 所示。



图 7.16　字符串中不存在子串

实现过程

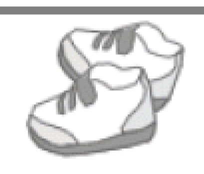
创建 index.php 文件，定义字符串变量，将该变量与通过 POST 方式传递进来的数据利用 stristr()函数进行比较并输出结果。具体代码如下：

```
<?php
$a = "Hello World !!";                                                    //定义字符串
 echo $a;                                                                 //输出字符串
 echo "<form action="method='post'>";                                     //输出表单
 echo "<input type='text' name='text' value='输入文本内容'>";
 echo "<input type='submit' name='sub' value='提交'>";
 echo "</form>";
     if(!empty($_POST["sub"])){                                           //通过 POST 方式传递参数
          if(stristr($a, $_POST["text"]) != ""){                          //判断是否存在子串
                echo "<script>alert('文本存在指定子串');</script>";             //提示
         }else{
                     echo    "<script>alert('文本不存在指定子串');</script>";   //提示
       }
     }
?>
```

技术要点

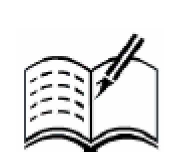
通过 stristr()函数可以判断一个字符串是否为另一个字符串的子串。其语法如下：

```
string stristr ( string haystack, string needle );
```



获取指定字符串（A）在另一个字符串（B）中首次出现的位置到（B）字符串末尾的所有字符串。参数 haystack 指定查找的字符串，参数 needle 指定查找的对象。

该函数如果执行成功，则返回剩余的字符串，否则将返回 false。



多学两招：

本函数还可以作为分割函数的一部分，从符合条件的字符串开始截取，一直取到文件末尾。该函数不区分大小写，如果要在区分字母大小写的情况下进行搜索，可以应用 strstr()函数。

实例 146　合并数组

（实例位置：配套资源\SL\07\146）

实例说明

数组与数组之间是彼此独立的，但是两个数组可以合并到一起成为一个新数组。本实例通过数组函数 array_merge()实现合并数组，运行结果如图 7.17 所示。

合并数组

数组1：Array ([编程] => PHP范例宝典 [soft] => 明日科技 [456] => mingri)

数组2：Array ([0] => PHP编程 [1] => JAVA编程 [2] => C#编程 [3] => ASP.NET编程)

合并后的数组Array ([编程] => PHP范例宝典 [soft] => 明日科技 [0] => mingri [1] => PHP编程 [2] => JAVA编程 [3] => C#编程 [4] => ASP.NET编程)

图 7.17　合并数组

实现过程

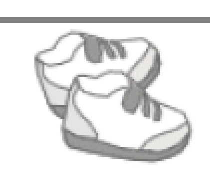
创建 index.php 文件，定义两个数组变量并为数组变量赋值。当用户单击“合并数组”按钮时，利用函数 array_merge 合并数组，并打印合并后的数组。其代码如下：

```
<?php
$array =array("编程"=>"PHP 实例宝典","soft"=>"明日科技","456"=>"mingri"); //定义数组 1
echo "数组 1：";
print_r($array);                                          //打印数组
echo "<br><br>";
$arr = array("PHP 编程","JAVA 编程","C#编程","ASP.NET 编程");   //定义数组 2
echo"数组 2：";
print_r($arr);                                            //打印数组
echo "<br><br>";
  if(empty($_POST["sub"])){                               //通过 POST 方式传递参数
        echo "合并后的数组";
        print_r(array_merge($array,$arr));                //打印合并后的数组
  }
?>
```

技术要点

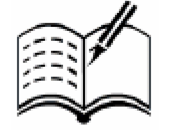
数组的合并是应用了 array_merge 函数。其语法如下：

```
array array_merge ( array array1 [, array array2 [, array ...]] );
```



参数说明：

- ☑ array array1：数组 1。
- ☑ array array2：数组 2。



多学两招：

array_merge() 将一个或多个数组的单元合并起来，一个数组中的值附加在前一个数组的后面。返回作为结果的数组。如果输入的数组中有相同的字符串键名，则该键名后面的值将覆盖前一个值。然而，如果数组包含数字键名，后面的值将不会覆盖原来的值，而是附加到后面。如果只给了一个数组并且该数组是数字索引的，则键名会以连续方式重新索引。

实例 147　拆分数组

（实例位置：配套资源\SL\07\147）

实例说明

存在数组的合并函数当然也就存在数组的拆分函数。本实例通过函数 array_chunk()实现数组的拆分，运行结果如图 7.18 所示。

拆分数组

原数组：Array ([0] => PHP编程 [1] => JAVA编程 [2] => C#编程 [3] => ASP.NET编程)

拆分后的数组：Array ([0] => Array ([0] => PHP编程 [1] => JAVA编程) [1] => Array ([0] => C#编程 [1] => ASP.NET编程))

图 7.18　拆分数组

实现过程

创建 PHP 脚本文件，定义数组变量并为此数组变量赋值。当用户单击“拆分数组”按钮时，打印被拆分后的数组。其代码如下：

```
<?php
$arr = array("PHP 编程","JAVA 编程","C#编程","ASP.NET 编程");    //定义数组
echo"原数组：";
print_r($arr);                                                  //打印数组
echo "<br><br>";
echo "拆分后的数组：";
if(empty($_POST["sub"])){                                       //通过 POST 方式传递参数
        print_r(array_chunk($arr,2))."<br>";                    //拆分数组
  }
?>
```

技术要点

array_chunk()函数可以将一个数组分割成多个，原理就是将原有数据根据传递的参数不同分割成多个二维数组。其语法如下：

```
array array_chunk ( array input, int size [, bool preserve_keys] );
```



参数说明：

- ☑ array input：源数组。
- ☑ int size：分割成指定数量的数组。

Note

多学两招：

在用 array_chunk()函数拆分的数组中，最后一个数组的单元数目可能会少几个。

实例 148 将时间和日期转换为时间戳

（实例位置：配套资源\SL\07\148）

实例说明

将日期转换为时间戳是将时间进行数学运算的重要手段。本实例通过 time()函数实现将日期和时间转换为时间戳，运行结果如图 7.19 所示。

当前时间：2010-06-26 10:16:16
转换成时间戳为：1277518576

图 7.19 将日期和时间转换为时间戳

实现过程

创建 index.php 文件。首先设置 PHP 的时区为中国上海，然后利用 time()函数获取当前的时间戳，并通过 echo 语句输出时间戳。其代码如下：

```
<?php
date_default_timezone_set("Asia/ShangHai");          //设置时区
echo "当前时间：".date('Y-m-d H:i:s');                //取得当前时间
echo "<br>转换成时间戳为：".time();                   //转换为时间戳
?>
```

技术要点

获取系统 UNIX 时间戳的方法有很多种，time()函数是比较常用的一种方法，其语法如下：

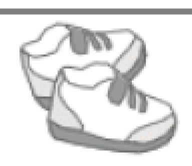
```
int time ( void );
```

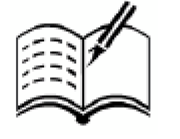
该函数没有参数，返回值为 UNIX 时间戳的整数值。

多学两招：

将当前日期和时间转换为时间戳的方法有很多种，下面笔者就通过 strtotime()函数实现上例功能。代码如下：

```
<?php
    date_default_timezone_set("Asia/ShangHai");        //设置时区
    echo "当前时间：".date('Y-m-d H:i:s');              //设置当前日期和时间
    echo "转换时间戳为："strtotime("now");              //转换为时间戳
?>
```



实例 149　网页闹钟

（实例位置：配套资源\SL\07\149）

实例说明

闹钟是我们生活中经常使用的一个小工具，它会在指定时间叫你起床。在 Web 程序中，我们也可以应用这个原理，开发一个“网页闹钟”，提示用户在指定的时间或者时间段内要做什么工作。本实例通过 strtotime()函数实现网页闹钟程序，运行结果如图 7.20 所示。

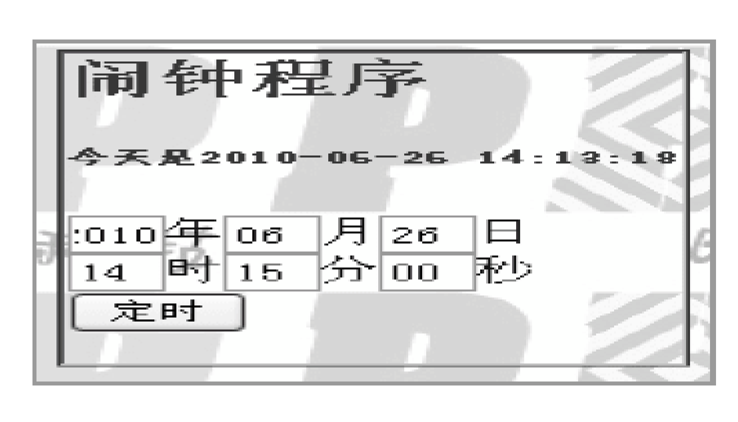


图 7.20　网页闹钟

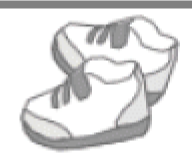
实现过程

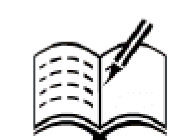
具体步骤如下：

（1）创建 index.php 文件。首先定义 PHP 时区为中国上海，然后编写 form 表单，并通过 mktime()函数实现获取当前的时间戳。其代码如下：

```
<?php
echo "<table border='1' bordercolor='#FF0000' cellspacing='0' align='center'><tr><td>"; //输出表格
echo "<h2 style='color:#0033FF'>闹钟程序</h2>";                    //输出标题
date_default_timezone_set("Asia/ShangHai");                        //设置时区
echo "<h6 style='color:red'>今天是".date("Y-m-d H:i:s")."</h6>";      //输出当前时间
echo "<form action='' method='post'>";                             //输出表单
echo "<input type='text' name='text1' size='2'>年<input type='text' name='text2' size='2'>月";
echo "<input type='text' name='text3' size='2'>日<br><input type='text' name='text4' size='2'>时";
echo "<input type='text' name='text5' size='2'>分<input type='text' name='text6' size='2'>秒<br>";
echo "<input type='submit' name='sub' value='定时'>";
echo "</form>";
$a = $_POST["text1"];                                              //取得文本框内容
$b = $_POST["text2"];                                              //取得文本框内容
$c = $_POST["text3"];                                              //取得文本框内容
$d = $_POST["text4"];                                              //取得文本框内容
$e = $_POST["text5"];                                              //取得文本框内容
$f = $_POST["text6"];                                              //取得文本框内容
$time = date(mktime($d, $e, $f, $b, $c, $a));                      //获取时间戳
$_SESSION['time'] = $time;                                         //保存在 SESSION 中
if(isset($_POST["sub"])){                                          /通过 POST 传递参数
  echo"<script>window.location.href='in.php';</script>";           //跳转
}
echo "</td></tr></table>";
?>
```

（2）创建 in.php 文件，应用 SESSION 实现特定日期的判定。如果当前日期已到指定的日期，则给出提示，否则输出“距离 XXX 的生日还有 X 天”字样。其代码如下：





```
<?php
date_default_timezone_set("Asia/ShangHai");                              //设置时区
$time_1 = time();                                                        //取得当前时间戳
if($_SESSION['time'] < $time_1){                                         //判断条件
        echo "<script>alert('时间一去不复返');location.href='index.php'</script>";      //输出提示
}else{
  if($_SESSION['time'] == $time_1){                                      //判断条件
        echo "<script>alert('今天是 XXX 生日');</script>";                //输出提示
  }else{                                                                 //输出提示
        echo "距离 XXX 的生日还有".ceil((($_SESSION['time']-$time_1)/(3600*24)))."天";
  }
}
?>
```

技术要点

PHP 中应用 mktime()函数将一个时间转换成 UNIX 的时间戳值。mktime()函数根据给出的参数返回 UNIX 时间戳。时间戳是一个长整数，包含从 UNIX 纪元到给定时间的秒数。mktime()函数的语法如下：

```
int mktime ( [int hour [, int minute [, int second [, int month [, int day [, int year [, int is_dst]]]]]] );
```

mktime()函数的参数说明如表 7.1 所示。

表 7.1　mktime()函数的参数说明

参　　数	说　　明
hour	小时数
minute	分钟数
second	秒数（一分钟之内）
month	月份数
day	天数
year	年份数，可以是两位或四位数字，0～69 对应于 2000～2069，70～100 对应于 1970～2000
is_dst	参数 is_dst 在夏令时可以设为 1，如果不是则设为 0；如果不确定是否为夏令时则设为-1（默认值）

多学两招：

PHP 为 UNIX 时间戳的处理提供各种函数。到目前的 PHP 版本为止，由于任何已知 Windows 版本以及一些其他系统均不支持负的时间戳，因此在 Windows 中无法表示 1970 年 1 月 1 日之前的时间。因为目前 UNIX 时间戳是以 32 位二进制表示的，32 位二进制数值范围为(-2147483648~+2147483647)，因此，目前 UNIX 时间戳可表示的最大时间为 2038 年 1 月 19 日 3 点 14 分 7 秒，该时刻时间戳为 2147483647，对于该时刻之后的时间，需要扩展表示 UNIX 时间戳的二进制位数。



PHP 数组应用

本章读者可以学到如下实例：

- ⏭ 实例 150　查看最便宜的图书
- ⏭ 实例 151　随机抽取图书
- ⏭ 实例 152　车牌摇号
- ⏭ 实例 153　获取上传文件的数据
- ⏭ 实例 154　$_FILES[]全局数组在文件上传中的应用
- ⏭ 实例 155　图书信息逆向输出
- ⏭ 实例 156　统计图书的数量
- ⏭ 实例 157　获取图书馆中最受欢迎的 3 本图书
- ⏭ 实例 158　生成在线考试题
- ⏭ 实例 159　向购物车中添加商品
- ⏭ 实例 160　查看购物车
- ⏭ 实例 161　从购物车中移去指定商品
- ⏭ 实例 162　修改商品购买数量
- ⏭ 实例 163　清空购物车
- ⏭ 实例 164　收银台结账





实例 150　查看最便宜的图书

（实例位置：配套资源\SL\08\150）

实例说明

在图书管理系统中，将图书的名称和价格存放在数组中，并按价格由高到低的顺序排列。当需要查找最便宜的图书时，单击“查看最便宜的图书”按钮，应用 array_pop()函数弹出数组中末尾的单元，运行结果如图 8.1 所示。

图 8.1　检索最便宜的图书

实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计页面表单元素。

（2）编写实现代码，具体内容如下：

```
<?php
$array = array("79"=>"jsp 图书","69"=>"asp 图书","59"=>"php 图书");        //声明一个数组$array
if($_POST){                                                             //通过$_POST[]方式传递表单的数组$array
  $a = array_pop($array);
  echo "最便宜的图书是：$a";
}
?>
```

技术要点

本实例主要使用 array_pop()函数输出并返回数组的最后一个单元，然后将数组的长度减一，如果数组为空（或者不是数组）则返回 null。该函数的语法格式如下：

```
mixed array_pop ( array array)
```

参数 array 为输入的数组。

实例 151　随机抽取图书

（实例位置：配套资源\SL\08\151）

实例说明

本实例应用 array_rand()函数从图书馆的技术类图书中随机抽取 3 本书做推荐图书，运行结果如图 8.2 所示。

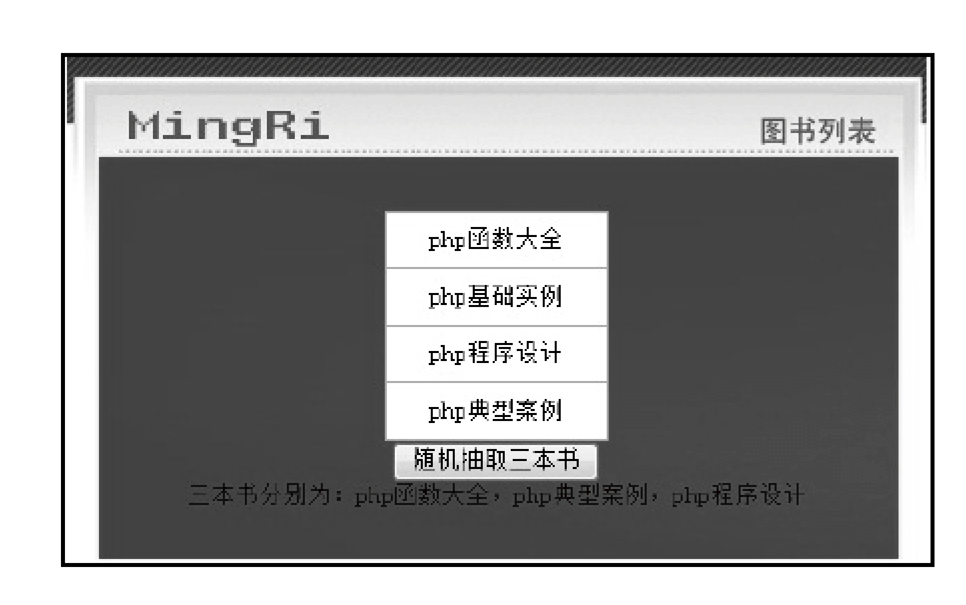
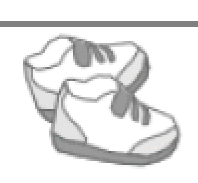


图 8.2　随机抽取图书

实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计表单



页面。

（2）array_rand()函数返回的是随机抽取的单元键名，并不是单元值。本实例的关键代码如下：



```
<?php
$array = array("php 函数大全","php 基础实例","php 程序设计","php 典型案例");
if($_POST){
    $result = array_rand($array, 3);
    echo "三本书分别为：".$array[$result[0]]."，".$array[$result[1]]."，".$array[$result[2]];
}
?>
```

技术要点

本实例主要使用了 array_rand()函数，该函数主要用于从数组中随机取出一个或多个单元。如果只取出一个，则返回一个随机单元的键名；否则，返回一个包含随机键名的数组。这样就可以随机从数组中取出键名和值。array_rand()函数的语法格式如下：

```
mixed array_rand ( array input [, int num_req]);
```

参数说明：

☑ input：必选参数，输入的数组。

☑ num_req：可选参数，指明想取出多少个单元，如果没有指定，默认为 1。

实例 152　车牌摇号

（实例位置：配套资源\SL\08\152）

实例说明

我们可以把随机抽取数组中的元素看成一个彩票抽奖的大摇箱，在原理上两者确是没有什么区别的，但是彩票抽奖的大摇箱可能存在一点点的偶然性，例如，大摇箱里的球弹力不同等。本实例通过随机函数 rand()实现随机抽取数组中存储的车牌号码，运行结果如图 8.3 所示。

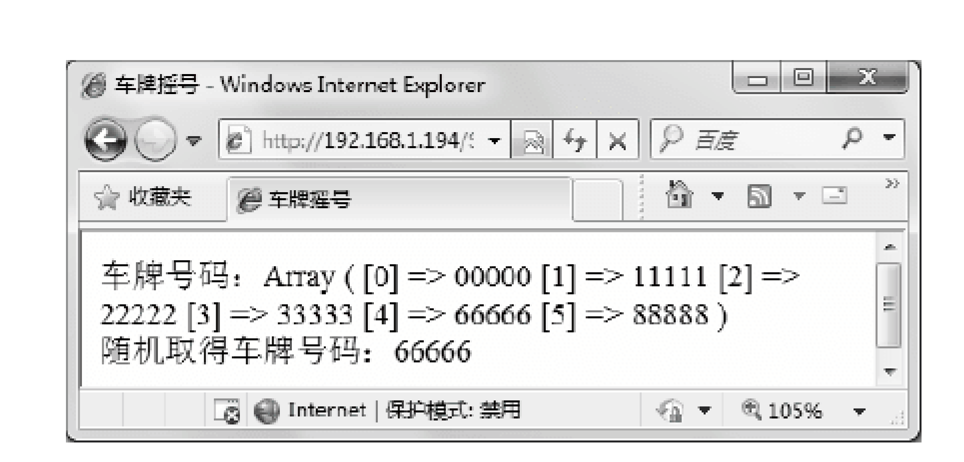


图 8.3　车牌摇号

实现过程

创建 PHP 脚本。首先定义数组变量并为数组变量赋值，然后利用 rand()函数获取 0～4 中的一个随机数，并将此随机数赋给数组变量作为数组变量的键值，最后输出此数组变量。其代码如下：

```
<?php
  $numbers = array('00000','11111','22222','33333','66666','88888');
  echo "车牌号码：";
  print_r($numbers);
```



```
    $rand = rand(0,6);
    echo "<br>随机取得车牌号码：".$numbers[$rand];
?>
```

技术要点

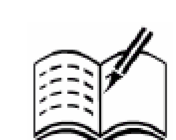
随机获取数组中的元素，可以用 rand()函数获取随机数值，并将随机数值指向数组下标，从而实现随机获取指定元素的值。rand()函数的语法如下：

```
int rand ( [int min, int max]);
```

参数说明：

☑ int min：随机数的最小值。

☑ int max：随机数的最大值。



实例 153　获取上传文件的数据

（实例位置：配套资源\SL\08\153　视频位置：配套资源\SP\08\153）

实例说明

获取上传文件的数据，需要使用$_FILES[]全局数组中的$_FILES["file"]["name"]来实现。本实例使用该全局数组获取上传文件的名称，运行结果如图 8.4 所示。

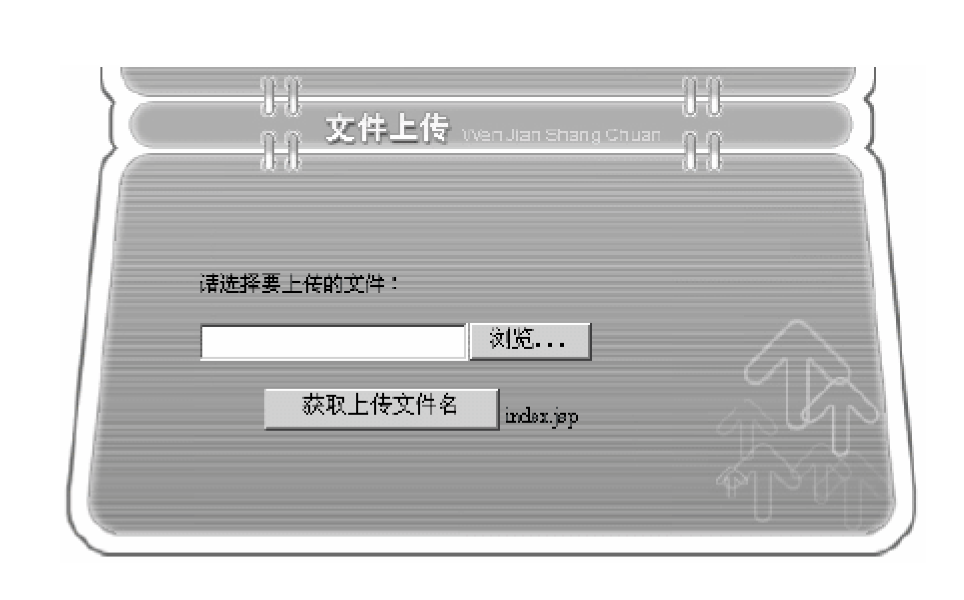


图 8.4　获取上传文件的数据

实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计表单页面。

（2）编写实现代码，具体内容如下：

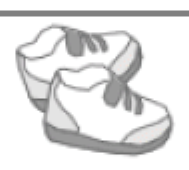
```
<?php
if(isset($_POST['Submit']) and $_POST['Submit']=="获取上传文件名"){    //判断按钮的值是否存在，是否为空
    echo $_FILES["file"]["name"];                                      //输出上传文件名
}
?>
```

技术要点

本实例主要使用了$_FILES[]全局数组中的$_FILES["file"]["name"]，与其他全局数组不同，$_FILES[ ]全局数组为一个多维数组，该数组用于获取通过 POST 方式上传文件时的相关信息。如果为单文件上传，则该数组为二维数组，如果为多文件上传，则该数组为三维数组。下面对该数组的具体参数取值进行介绍。

☑ $_FILES["file"]["name"]：从客户端上传的文件名称。

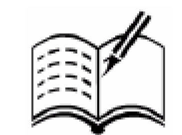
☑ $_FILES["file"]["type"]：从客户端上传的文件类型。




☑ $_FILES["userfile"]["size"]：已上传文件的大小。

☑ $_FILES["file"]["tmp_name"]：文件上传到服务器后，在服务器中的临时文件名。

☑ $_FILES["file"]["error"]：返回在上传过程中发生错误的错误代号。



Note

实例 154　$_FILES[]全局数组在文件上传中的应用

（实例位置：配套资源\SL\08\154　视频位置：配套资源\SP\08\154）

实例说明

在开发文件上传的程序时，为了保证上传文件的安全、合理，需要对上传文件的大小、类型和名称等进行限制，同时，还要在上传失败时给出一个正确的错误提示信息。所有这些操作都可以通过$_FILES[ ]全局数组来完成。

开发一个单文件上传的实例，并应用$_FILES[]全局数组获取上传文件在客户端的名称和在服务器端的临时名称，同时根据$_FILES[]全局数组返回的错误代码给出正确的错误提示信息。其运行效果如图 8.5 所示。



图 8.5　获取上传文件的数据

实现过程

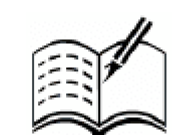
具体步骤如下：

（1）创建 index.php 文件。首先添加表单，设置表单的 enctype 属性值为“multipart/form-data”。然后设置 action 属性值为 POST，并添加隐藏域限制上传文件的大小、添加文件域选择上传文件。最后添加提交按钮，将数据提交到本页。

（2）通过$_FILES[ ]全局数组获取上传文件的相关信息，通过 is_dir()和 mkdir()判断、创建上传文件存储的目录，通过 time()和 strstr()函数定义上传文件在服务器中的名称，通过 is_uploaded_file()函数判断文件是否为 HTPP POST 上传，通过 move_upload_file()函数完成文件上传。具体代码如下：

```
<?php
if(!empty($_FILES['up_picture']['name'])){                    //判断上传内容是否为空
  if($_FILES['up_picture']['error']>0){                       //判断文件是否可以上传到服务器
        echo "上传错误:";
  switch($_FILES['up_picture']['error']){//根据$_FILES[ ]全局数组返回的错误代码，输出不同的错
```





```
误信息
                case 1:
                    echo "上传文件大小超出配置文件规定值";
                break;
                case 2:
                    echo "上传文件大小超出表单中约定值";
                break;
                case 3:
                    echo "上传文件不全";
                break;
                case 4:
                    echo "没有上传文件";
                break;
            }
        }else{
            if(!is_dir("./upfile/")){                          //判断指定目录是否存在
                mkdir("./upfile/");                            //创建目录
            }
            $path='./upfile/'.time().strstr($_FILES['up_picture']['name'],'.');//定义上传文件名称和存储位置
            if(is_uploaded_file($_FILES['up_picture']['tmp_name'])){//判断文件是否为 HTPP POST 上传
                if(!move_uploaded_file($_FILES['up_picture']['tmp_name'],$path)){ //执行上传操作
                    echo "上传失败";
                }else{
                echo "文件".$_FILES['up_picture']['name']."上传成功，大小为：".$_FILES['up_picture']
['size'];
                }
            }else{
                echo "上传文件".$_FILES['up_pictute']['name']."不合法！ ";
            }
        }
    }
    ?>
```

技术要点

本实例对 PHP 的各种知识进行了整合应用，包括流程控制语句（if 和 switch）、数组（$_FILES[]全局数组）、文件系统函数（id_dir()、mkdir()、is_uploaded_file()和 move_uploaded_file()）、日期时间函数（time()）和字符串函数（strstr()）。

实例 155　图书信息逆向输出

（实例位置：配套资源\SL\08\155）

实例说明

在图书列表中，将 php 书籍以列表的形式随机存储到数组中，如图 8.6 所示。单击“逆序排列”按钮，应用 arsort()函数按图书名称进行逆向排序（由高到低），获取的结果列表如图 8.7 所示。




Note

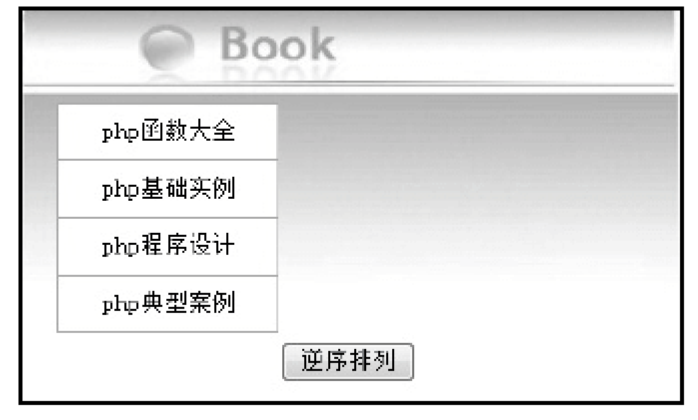


图 8.6　php 图书列表

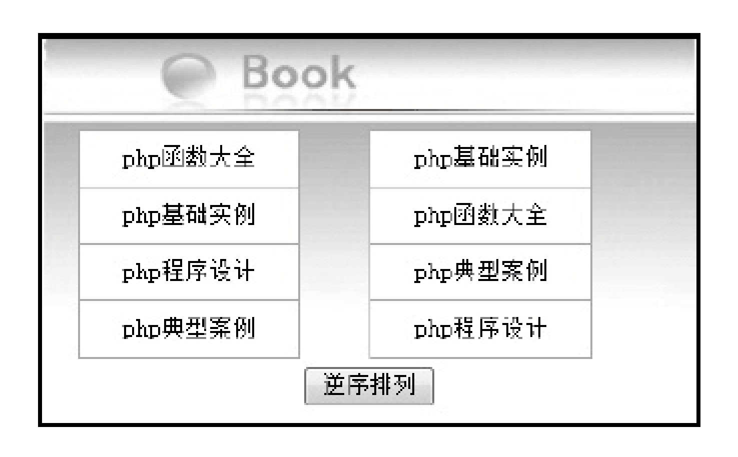


图 8.7　逆向排序后的图书列表

实现过程

arsort()函数排序的对象是数组中的值，并保持索引和其对应值的关联关系，代码如下：

```
<?php
$array = array("php 函数大全","php 基础实例","php 程序设计","php 典型案例");
if($_POST){
	arsort($array);
	echo '<table border="0" cellpadding="0" cellspacing="1" bgcolor="#AAAAAA">';
	foreach($array as $value){
		echo '<tr align="center">
				<td width="120" height="30" bgcolor="#FFFFFF">'.$value.'</td>
			</tr>';
	}
	echo '</table>';
}
?>
```

技术要点

对数组进行逆向排序，数组的索引保持和单元的关联，主要用于对那些单元顺序很重要的数组进行逆向排序。

语法如下：

```
void arsort ( array array [, int sort_flags])
```

参数说明：

☑ array：必选参数，输入的数组。

☑ sort_flags：可选参数，可改变排序的行为，排序类型标记。

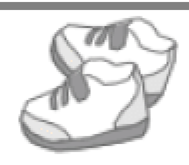
- SORT_REGULAR：正常比较单元。
- SORT_NUMERIC：单元被作为数字来比较。
- SORT_STRING：单元被作为字符串来比较。

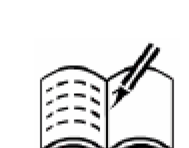
实例 156　统计图书的数量

（实例位置：配套资源\SL\08\156）

实例说明

在图书统计列表页面中，将图书的数据存放在数组中，并应用 count()函数对数组中的





图书数量进行统计，单击“统计图书”按钮，即可输出图书的总数量，运行结果如图 8.8 所示。

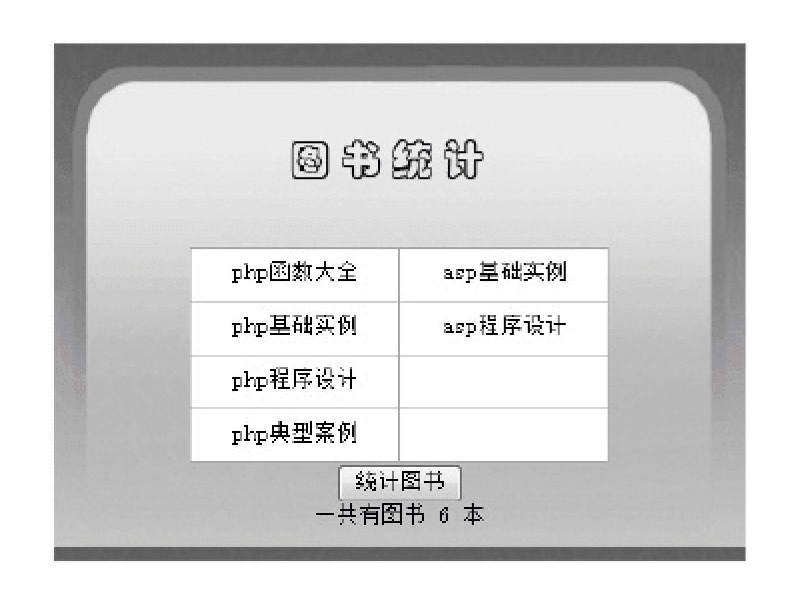
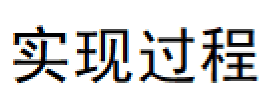


图 8.8　统计图书的数量



实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计表单页面。

（2）编写实现代码，本实例的关键代码如下：

```
<?php
$array = array("php 函数大全","php 基础实例","php 程序设计","php 典型案例","asp 基础实例","asp 程序设计");
if($_POST){
	$result = count($array);
	echo "一共有图书 $result 本";
}
?>
```

技术要点

count()函数用于返回数组中的单元数目，用来计算数组中值的个数。该函数的语法格式如下：

```
int count ( mixed array [, int mode])
```

参数说明：

☑ array：必选参数，输入的数组。

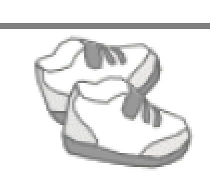
☑ mode：可选参数，COUNT_RECURSIVE（或 1），如选中此参数，本函数将递归地对数组计数，对计算多维数组的所有单元尤其有用，此参数的默认值为 0。

实例 157　获取图书馆中最受欢迎的 3 本图书

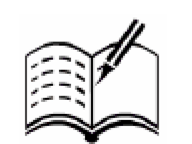
（实例位置：配套资源\SL\08\157）

实例说明

本实例将图书馆内 php 书籍的列表及借阅人数随机存放到数组中，应用 krsort()函数对图书进行逆向排序，然后返回排序好的数组列表，再应用 array_values()函数取出数组的值



并重设数字索引，因为 list()函数仅用于数字索引并从 0 开始，最后应用 list()函数获取最受读者欢迎的 3 本 php 图书，如图 8.9 所示。





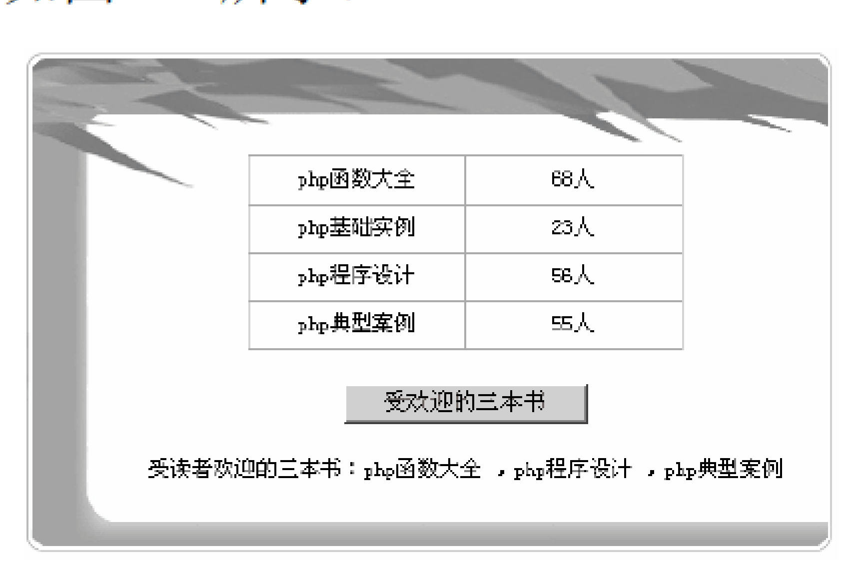


图 8.9　获取图书馆中最受欢迎的 3 本书

实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计表单页面。

（2）编写实现代码，本实例的关键代码如下：

```
<?php
$array = array("68 人"=>"php 函数大全","23 人"=>"php 基础实例","56 人"=>"php 程序设计","55 人"=>"php 典型案例");
if($_POST){
  krsort($array);                          //数组排序
  $result = array_values($array);          //获取数组的值
  list($first,$second,$third) = $result;   //遍历数组
  echo "受读者欢迎的三本书：$first ， $second ， $third";
}
?>
```

技术要点

把数组中的值赋给一些变量。与 array()函数类似，这不是真正的函数，而是语言结构。list()函数仅能用于数字索引的数组，且数字索引从 0 开始。

语法如下：

```
void list ( mixed ...)
```

参数 mixed 为被赋值的变量名称。

实例 158　生成在线考试题

（实例位置：配套资源\SL\08\158）

实例说明

考试题的添加以字符串的形式进行提交，每个考试题之间以*进行分隔，然后应用 explode()函数将字符串转换成数组，最后通过 for 循环语句和 count()函数完成考试题的输出，运行结果如图 8.10 所示。

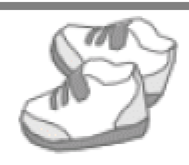
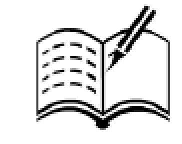




图 8.10　考试题输出



实现过程

具体步骤如下：

（1）创建 index.php 文件，并设计表单页面。

（2）编写实现代码，具体内容如下：

```
<?php
if($_POST[Submit]!=""){
  $content=$_POST[content];                  //获取表单提交的数据
  $data=explode("*",$content);              //explode 函数对数据进行拆分，以*号为分隔符
  for($i=0;$i<count($data);$i++){           //for 循环输出数组中的数据
        echo $data[$i]."<br><br>";          //输出元素值
  }
 }
?>
```

技术要点

本实例主要使用了 explode()函数和 count()函数。

（1）explode()函数。

explode()函数的作用是使用指定的分隔符来分割字符串，同时将分割后的字符串存成数组并返回。

语法如下：

```
array explode ( string separator, string string [, int limit] )
```

参数说明：

☑ separator：是要使用的分隔符，可以是单个字符，也可以是一个字符串。

☑ string：表示的是要分割的字符。

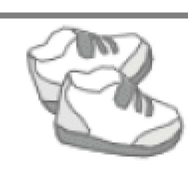
☑ limit：表示要分割的个数，如果 limit 参数是负数，则返回除了最后的 limit 个元素外的所有元素。

（2）count()函数。

count()函数返回数组中的单元数目，用来计算数组中值的个数。

语法如下：

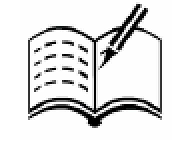
```
int count ( mixed array [, int mode])
```



参数说明：

☑ array：必选参数，输入的数组。

☑ mode：可选参数，COUNT_RECURSIVE（或 1），如选中此参数，本函数将递归地对数组计数，对计算多维数组的所有单元尤其有用，此参数的默认值为 0。



Note

多学两招：

电子商务系统中的购物车同实际生活中的购物车一样，都是用于暂时保存挑选的商品。购物车主要包括所选商品的添加、查看购物车、商品购买数量的修改、从购物车移去指定商品、清空购物车 5 部分。用户单击商品展台中的“放入购物车”超链接，可以将对应的商品添加至购物车。在查看购物车页面中，单击“移除”超链接可以从购物车中移去指定商品；在“数量”文本框中输入购买数量后，单击“更改商品数量”按钮即可修改指定商品的购买数量；单击“清空购物车”超链接，将退回购物车中的全部商品；如果用户确认购买当前购物车中的全部商品，可以单击“去收银台”超链接，进行订单处理。

下面通过具体的实例 159～实例 164 来向读者讲解如何应用 PHP 数组和 SESSION 变量共同实现购物车的全部功能。

实例 159　向购物车中添加商品

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

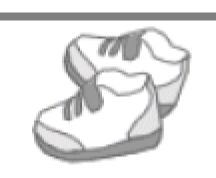
“添加至购物车”页主要用于将商品信息暂时保存到购物车中，而且在一些电子商务网站中，它是客户端程序中非常关键的一个功能，主要用来帮助用户完成商品的选购。本实例中的购物车是采用自定义函数 cart()来存储购物数据的，购物信息被保存在 SESSION 中。运行本实例，用户单击“放入购物车”按钮，即可将商品添加到购物车中，运行结果如图 8.11 所示，用户可以单击“首页”超链接继续选购商品。本实例中，对于同一商品的多数量选购，可以直接在购物车中对其数量进行更改，而不可以多次将商品放入购物车中。

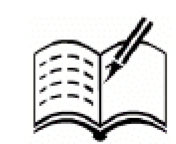


图 8.11　向购物车中添加商品

实现过程

用户看好一件商品但又没有确定买不买时，就可以将商品先放进购物车中。下面将介





绍如何将商品添加至购物车。

（1）当用户登录到网站时，系统将为每个用户分配两个 SESSION 变量$producelist 和$quatity，分别用来存储用户放入购物车中的商品 ID 和商品的数量。一个变量同一时刻只能有一个值，那么如何将多个 ID 值同时保存在一个$producelist 变量中呢？首先将 ID 转变成字符型变量，并且将这些变量用字符“@”进行连接，比如用户分别将 ID 为 1、3、5 的商品放入购物车中，这时 SESSION 变量$producelist 的值应该为“1@3@5@”。然后，用函数 explode()完成 ID 的提取，并且判断用户提交的 ID 值在$producelist 变量中是否存在，如果存在则给出提示信息，否则可以执行添加操作。addgouwuche.php 的代码如下：

```
<?php
session_start();                                    //开启会话
$id=strval($_GET["id"]);                            //获取商品 ID 值
//将 SESSION 变量$producelist 中的内容用字符“@”进行分割，并将结果保存在数组$array 中
$array=explode("@",$_SESSION["producelist"]);
for($i=0;$i<count($array)-1;$i++) {                 //循环输出数组中的元素
    if($array[$i]==$id){//如果$array 数组中存在与$id 相等的元素，说明该$id 所对应的商品已经在购物车中
      echo "<script>alert('该商品已经在您的购物车中!');history.back();</script>";
      exit;
    }
}
//如果该商品不在购物车中，则将该商品的 ID 值连接到 SESSION 变量$producelist 之后，并用“@”进行分割
$_SESSION['producelist']=$_SESSION['producelist'].$id."@";
//同时将该商品的数量用“@”进行分割保存在 SESSION 变量$quatity 中，并将默认数量设置为 1
$_SESSION['quatity']=$_SESSION['quatity']."1@";
header("location:shopcar.php");     //添加成功之后将该页定位到 shopcar.php 页面显示购物车中的内容
?>
```

技术要点

将商品添加至购物车其关键是应用会话创建 SESSION 变量，为用户分配购物车，然后应用 explode()函数向购物车中添加数据，最后通过 for 语句和 count()函数检测购物车中是否存在用户提交的商品。

（1）通过 session_start()函数初始化 SESSION 变量，通过$_SESSION[]直接为变量赋值。

（2）explode()函数，按照指定的规则对一个字符串进行分割，返回值为数组。其语法如下：

```
array explode(string separator,string str,[int limit])
```

explode()函数的参数说明如表 8.1 所示。

表 8.1　explode()函数的参数说明

设　置　值	描　　述
separator	必选参数，指定的分割符。如果 separator 为空字符串（""），explode()将返回 false。如果 separator 所包含的值在 str 中找不到，那么 explode()函数将返回包含 str 单个元素的数组



续表

设　置　值	描　　述
str	必选参数，指定将要被分割的字符串
limit	可选参数，如果设置了 limit 参数，则返回的数组包含最多 limit 个元素，而最后的元素将包含 string 的剩余部分；如果 limit 参数是负数，则返回除了最后的-limit 个元素外的所有元素

Note

（3）count()函数。

```
int count(mixed var)
```

count()函数的作用是用来计算变量 var 中元素的个数。如果变量为空，则返回 0；如果变量是数组，则返回数组元素的个数；如果是普通型变量，则返回 1。

指点迷津：

上面说到的 ID 指的是 tb_shangpin 表中的 ID 字段。

实例 160　查看购物车

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

用户选购完商品后，为了方便用户随时查看所选购的商品，可以通过查看购物车来查看当前用户购物车中的商品信息。在导航条中单击“购物车”超链接，即可打开“查看购物车”页面。运行结果如图 8.12 所示。



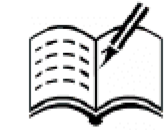
图 8.12　查看购物车

实现过程

在 shopcar.php 页面中，实现查看购物车功能，首先需要判断购物车是否为空，如果为空，需要将页面重定向到购物车首页面；否则显示购物车信息，主要将保存在 SESSION 中的购物信息应用 for 循环语句输出到浏览器中。其关键代码如下：

```
<?php
session_register("total");        //注册 SESSION 变量$total 用来保存所有商品价格总和
if($_GET['qk']=="yes"){           //判断用 GET 方法提交的 qk 的值是否为 yes，如是则使$producelist
和$quatity 的值为空串，才$_SESSION["producelist"]="";
```





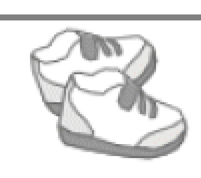

```
    $_SESSION["quatity"]="";
  }
  $arraygwc=explode("@",$_SESSION['producelist']);
  $s=0;                       //用$s 保存购物车中商品 ID 的总和
  for($i=0;$i<count($arraygwc);$i++){
    $s+=intval($arraygwc[$i]);
  }
  if($s==0 ) {                //如果$s 的值为空，则说明购物车中无商品
    echo "<tr>";
        echo" <td height='25' colspan='6' bgcolor='#FFFFFF' align='center'>您的购物车为空!</td>";
        echo"</tr>";
  }else {                     //如果$s 的值不为空，则显示购物车中所有商品信息
  ?>
    <! --省略部分代码 -->
  <?php
    $total=0;                                         //初始化$total 变量
    $array=explode("@",$_SESSION["producelist"]);     //分离购物车中的商品 ID
    $arrayquatity=explode("@",$_SESSION["quatity"]);  //分离购物车中的商品数量
    while(list($name,$value)=each($_POST)) {          //获取 POST 传递值
          for($i=0;$i<count($array)-1;$i++){
                if(($array[$i])==$name){
                    $arrayquatity[$i]=$value;
                }
          }
    }
    $_SESSION["quatity"]=implode("@",$arrayquatity);  //更新购物车中商品数量
    for($i=0;$i<count($array)-1;$i++) {               //循环输出
          $id=$array[$i];
          $num=$arrayquatity[$i];
          if($id!=""){
                $sql=mysql_query("select * from tb_shangpin where id='".$id."'",$conn);   //执行数据库操作命令
                $info=mysql_fetch_array($sql);               //执行数据库操作
                $total1=$num*$info["huiyuanjia"];            //计算会员价格
                $total+=$total1;                             //增加商品总价格
                $_SESSION["total"]=$total;       //将商品总价格赋值给$_SESSION["total"]变量
  ?>
  <! --省略部分代码 -->
  <?php
                }
    }
   ?>
```

技术要点

查看购物车的开发思路：首先判断购物车是否为空，如果为空，则显示“您的购物车为空!”的提示信息；否则，将 SESSION 数组中的购物信息显示到页面中。

（1）通过 explode()函数获取购物车中存储的商品 ID 和商品数量。

（2）通过 implode()函数更新购物车中商品的数量。



implode()函数，将数组中的元素组合成一个新字符串。其语法如下：

```
string implode(string glue, array pieces)
```

参数 glue 是字符串类型，指定分隔符；参数 pieces 是数组类型，指定要被合并的数组。

（3）创建$_SESSION["total"]变量，存储商品的总价数据。



实例 161　从购物车中移去指定商品

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

在购物过程中，顾客想将已经选择的商品退回到货物架上是商品选购时经常发生的事情。因此，在开发一个电子商城网站时，必须考虑到这一点，做好需求分析，从而使网站更加完善。那么本实例在设计购物车时又是如何实现从购物车中移去指定商品的呢？这就需要在查看购物车页面中添加一个将指定商品退回的功能，即从购物车中移去指定商品，运行结果如图 8.13 所示。



图 8.13　从购物车中移去指定商品

实现过程

具体步骤如下：

（1）在 shopcar.php 页面的“购物车”中添加“移除”超链接，链接到 removegwc.php 文件中，根据超链接传递的 ID 值完成对指定商品的删除操作。创建超链接的代码如下：

```
<a href="removegwc.php?id=<?php echo $info['id']?>">移除</a>
```

（2）在 removegwc.php 文件中，首先用函数 explode()将 SESSION 变量$producelist 以“@”进行分割，并把分割出的子串存放到数组中，然后将用户要删除的商品对应的数组元素赋予空值，最后将数组元素重新组合成新串，即完成指定商品的移除操作。其关键代码如下：

```
<?php
session_start();
$id=$_GET['id'];                                        //获取用户打算移去商品的 ID
$arraysp=explode("@",$_SESSION["producelist"]);
$arraysl=explode("@",$_SESSION["quatity"]);             //分别将购物车中的商品 ID 和对应商品的数
```





```
量存放到数组$arraysp 和$arraysl 中
        for($i=0;$i<count($arraysp);$i++) {
          if($arraysp[$i]==$id) {                    //通过循环寻找与 ID 值相等的数组元素
                $arraysp[$i]="";
                $arraysl[$i]="";                     //将与 ID 值相等的数组元素赋予空值
          }
        }
        $_SESSION["producelist"]=implode("@",$arraysp);
        $_SESSION["quatity"]=implode("@",$arraysl); //利用 implode()函数将数组元素重新组合成新串
        header("location:shopcar.php");              //重新定位到 shopcar.php 显示购物车
        ?>
```

技术要点

本实例实现从购物车中移去指定商品，主要应用 for 循环语句将用户指定商品的数组元素赋予空值，代码如下：

```
        for($i=0;$i<count($arraysp);$i++) {
            if($arraysp[$i]==$id) {                  //通过循环寻找与 ID 值相等的数组元素
             $arraysp[$i]="";
             $arraysl[$i]="";                        //将与 ID 值相等的数组元素赋予空值
          }
         }
```

实例 162　修改商品购买数量

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

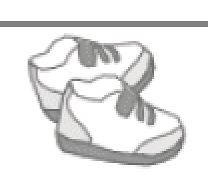
为了满足用户的不同需求，购物车中还需加入修改指定商品购买数量的功能。在购物车中，由于商品的数量被存放在文本框里，用户只需在某种商品后面的文本框中输入相应的数量即可。例如，在本实例中，将数码摄像机的数量改为 2 台，然后单击“更改商品数量”按钮，即可成功修改商品的购买数量，运行结果如图 8.14 所示。

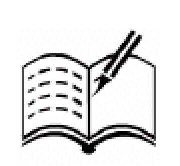


图 8.14　修改商品购买数量

实现过程

购物车中的商品默认数量是 1，如果用户打算购买多件相同的商品，就需要对购物车





Note

中商品数量变量的值进行修改。在 shopcar.php 文件中，修改商品购买数量是将购物车中某件商品对应的数组元素赋予新值。其关键代码如下：

```
while(list($name,$value)=each($_POST)) {                //提取表单中的商品 ID 和新数量
  for($i=0;$i<count($array)-1;$i++) {
    if(($array[$i])==$name) {
$arrayquatity[$i]=$value; //获取购物车中每种商品的数量，并将数量保存到$arrayquatity 数组中
  }
}
}
$_SESSION['quatity']=implode("@",$arrayquatity);        //将更新后的数据保存到购物车中
```

要实现购物车中商品数量的更新，前提是必须通过表单来提交指定商品的 ID 和要更新的数量，否则更新操作不会被执行。提交商品 ID 和更新数量的表单如下：

```
<form name="form1" method="post" action="shopcar.php">
  <input type="text" name="<?php echo $info['id'];?>" size="2" class="inputcss" value=<?php echo
$num;?>>
      <input name="submit" type="submit" class="buttoncss" value="更改商品数量">
</form>
```

在创建添加商品数量的表单元素时，以商品 ID 为表单元素的名称，以提交的商品数量为表单元素的值，最终将数据提交到 shopcar.php 页面中。

技术要点

本实例中通过 list()函数和 each()函数获取表单提交的商品 ID 和更改的商品数量。

list()函数将数组中的值赋给一些变量，该函数仅能用于数字索引的数组，且数字索引从 0 开始。其语法如下：

```
void list ( mixed ...)
```

参数 mixed 为被赋值的变量名称。

each()函数返回数组中的键名和对应的值，并向前移动数组指针。其语法如下：

```
array each ( array array)
```

参数 array 为输入的数组。

在语句“list($name,$value)=each($_POST)”中，变量$name 为指定商品的 ID，变量$value 为商品更新的数量值，而$_POST 则是表单提交的数据。

实例 163　清空购物车

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

在实例 162 中已经介绍了如何从购物车中移去指定的商品，本实例将介绍一种如何将购物车中全部商品一次性清空的方法。清空购物车的实现方法很简单，只需将保存在 SESSION 中的购物信息清空即可，运行结果如图 8.15 所示。

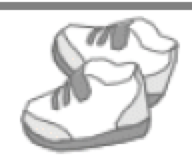
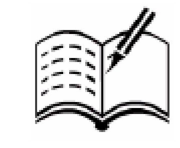




图 8.15　清空购物车



实现过程

在 shopcar.php 页面中，创建一个“清空购物车”的超链接，链接到 shopcar.php 文件中，设置超链接参数 qk 的值为 yes。超链接的代码如下：

```
<a href="gouwu1.php?qk=yes">清空购物车</a>
```

在 shopcar.php 页面中，判断超链接变量 qk 是否被设置，变量 qk 的值是否等于 yes，如果等于，则执行购物车的清空操作，即为$_SESSION["producelist"]和$_SESSION["quatity"]赋空值。其关键代码如下：

```
if(isset($_GET['qk']) && $_GET['qk']=="yes"){        //判断用户是否单击“清空购物车”
  $_SESSION["producelist"]="";                       //清空购物车中商品 ID
  $_SESSION["quatity"]="";                           //清空购物车中商品数量
}
```

技术要点

本实例主要是通过将保存商品 ID 和商品数量的 SESSION 变量清空，从而完成清空购物车的功能。在执行这个操作之前，应用 isset()函数判断超链接参数变量是否被设置，因为只有超链接参数变量被设置的情况下，才能执行清空操作，并且这只是其中的一个条件，同时还必须保证变量值是 yes 才能真正执行清空操作。

isset()函数，检查变量是否被设置，返回值为 true 或 false，由于这是一个语言结构而非函数，因此它无法被“变量函数”调用。其语法如下：

```
bool isset ( mixed var [, mixed var [, ...]] )
```

参数说明：

☑　var：必选参数，指定检测的变量。

☑　var2：可选参数，指定多个被检测的变量。

实例 164　收银台结账

（实例位置：配套资源\SL\08\159-164　视频位置：配套资源\SP\08\159-164）

实例说明

如同在超市中购物一样，将商品保存到购物车中并不是网上购物的最终目的，而到收

银台结账后，才算是一次购物过程的最终完成。前面所有功能都是为最后生成一个用户满意的订单做准备。生成订单时，不仅要保存用户订单中所购买的商品信息和订单信息，同时还需要返回一个订单号。用户单击查看购物车页面中的“去收银台”超链接即可进入到收银台结账页面填写订单信息。订单信息填写完成后，单击“提交订单”按钮，即可保存订单信息到数据表中，运行结果如图 8.16 和图 8.17 所示。



图 8.16　收银台结账

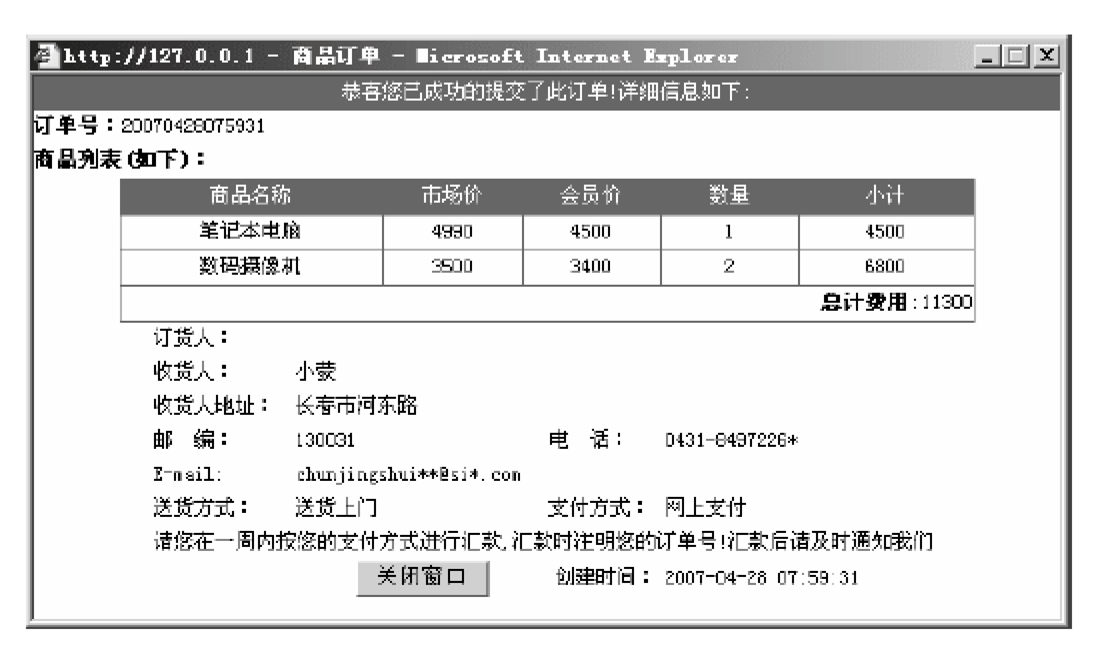


图 8.17　商品订单

实现过程

在 shopcar.php 页面中，单击“去收银台”超链接，将跳转到 cash_register.php 页面，填写订单信息，单击“提交订单”按钮将订单数据提交到 savedd.php 文件，将订单的概要信息保存到订单主表中，同时返回该订单编号。

（1）创建 cash_register.php 页面，生成提交订单信息的表单，将数据提交到 savedd.php 文件进行处理。表单的设计效果如图 8.18 所示。

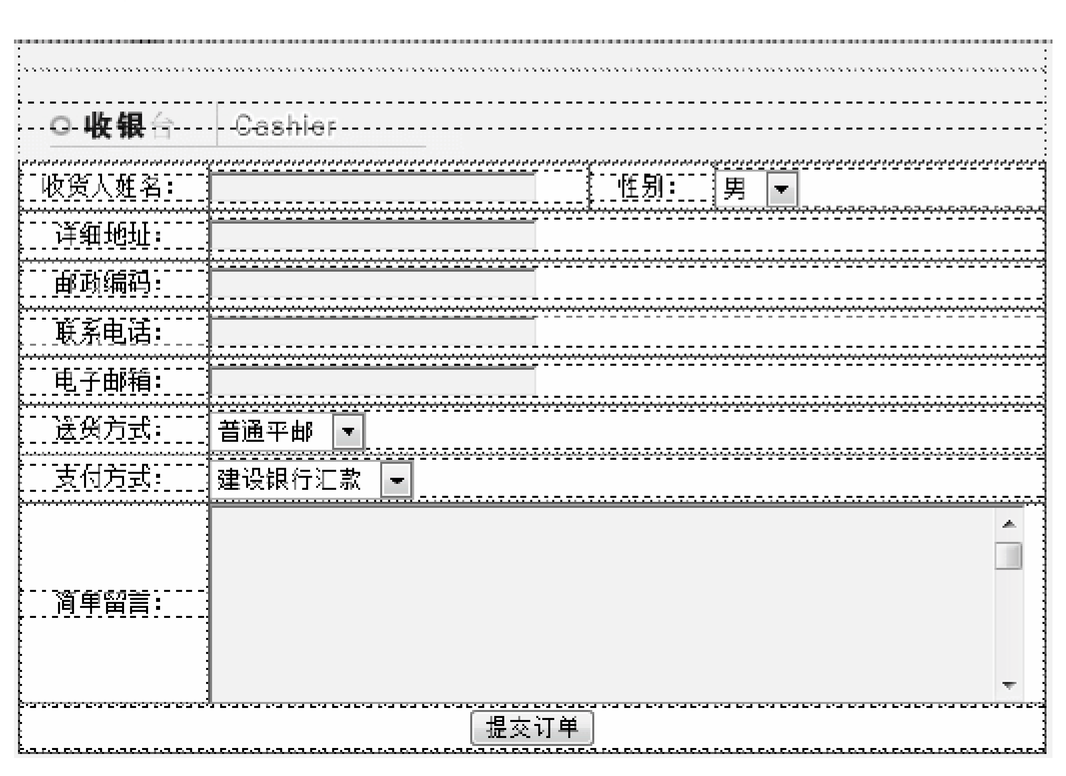


图 8.18　表单设计效果

（2）创建 savedd.php 文件，获取表单提交的订单数据，将其添加到指定的数据表中，并且生成订单号，数据添加成功后将重新定向到 cash_register.php 页面，将生成的订单号作为超链接的参数值进行传递。savedd.php 的关键代码如下：

```
<?php
session_start();                         //初始化 SESSION 变量
```





```
include("conn/conn.php");           //载入数据库连接文件
$dingdanhao=date("YmjHis");         //根据当前时间戳生成订单号
$spc=$_SESSION["producelist"];      //获取购物车数据
$slc= $_SESSION["quatity"];
//省略部分代码
if(trim($_POST["ly"])==""){
  $leaveword="";
}else{
        $leaveword=$_POST["ly"];
}
$xiadanren="Mrsoft";
$time=date("Y-m-j H:i:s");
$zt="未作任何处理";
$total=$_SESSION["total"];
mysql_query("insert  into  tb_dingdan(dingdanhao,spc,slc,shouhuoren,sex,dizhi,youbian,tel,email,shff,
zfff,leaveword,time,xiadanren,zt,total)  values  ('$dingdanhao','$spc','$slc','$shouhuoren','$sex','$dizhi','$youbian',
'$tel','$email','$shff','$zfff','$leaveword','$time','$xiadanren','$zt','$total')",$conn);          //执行添加语句
header("location:cash_register.php?dingdanhao=$dingdanhao");          //页面重定向
?>
```

（3）订单数据添加成功后，重定向到 cash_register.php 页面。在该页面中，首先判断订单号变量是否被设置，如果订单号被设置，则通过 JavaScript 脚本中的 open()方法打开 showdd.php 文件，输出生成的订单详细信息。cash_register.php 页面中的关键代码如下：

```
<?php
if(isset($_GET["dingdanhao"])){                        //判断订单号是否被设置
  $dd=$_GET["dingdanhao"];                             //获取订单号
      $array=explode("@",$_SESSION["producelist"]);    //读取购物车中数据
  $sum=count($array)*20+260;
      echo" <script language='javascript'>";      //通过 JavaScript 脚本中的方法打开 showdd.php 文
件，输出订单信息
  echo"
window.open('showdd.php?dd='+'".$dd."','newframe','top=150,left=200,width=600,height=".$sum.",menubar=n
o,toolbar=no,location=no,scrollbars=no,status=no ')";
  echo "</script>";
   }
?>
```

（4）创建 showdd.php 文件，并输出商品订单的详细信息。

技术要点

本实例中，订单号的生成应用的是 date()函数，以“YmjHis”格式获取当前的时间作为订单号。

在生成订单之后，要清空购物车中数据，即为$_SESSION["producelist"]和$_SESSION ["quatity"]赋空值。



日期和时间的处理

本章读者可以学到如下实例：

- ⏭ 实例 165　获取不同地区的当前时间
- ⏭ 实例 166　计算考试时间
- ⏭ 实例 167　输出中文格式的日期和时间
- ⏭ 实例 168　检验日期和时间的有效性
- ⏭ 实例 169　倒计时
- ⏭ 实例 170　判断时间的早晚
- ⏭ 实例 171　网页闹钟
- ⏭ 实例 172　计算程序的运行时间



Note

实例 165　获取不同地区的当前时间

（实例位置：配套资源\SL\09\165）

实例说明

系统的当前时间受时区限制。默认情况下，系统的当前时间是格林威治时间。用户正确取得所在的本地时间是很重要的。本实例通过 date()函数获取不同地区的当前时间，运行结果如图 9.1 所示。

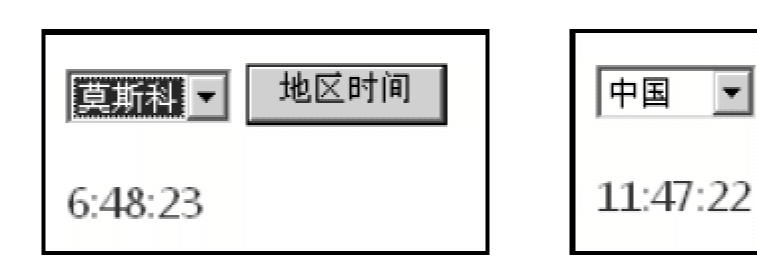


图 9.1　获取不同地区的当前时间

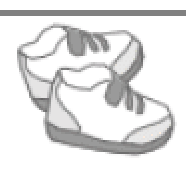
实现过程

新建 index.php 文件，当单击“地区时间”按钮时，首先为当前 PHP 服务时间设置时区，并通过 switch 语句根据地址栏传递参数的不同，通过 date()函数获取不同国家地区的时间，其代码如下：

```
<?php
    if($_POST['sub ']){                                  //通过 POST 方式传参
        date_default_timezone_set("Asia/ShangHai");      //设置时区
        switch($_POST['select ']){                       //条件语句
            case "中国";                                  //如果是中国
            echo date("H:i:s");                          //输出时间
            break;                                       //结束语句
            case "英国";                                  //如果是英国
            echo (date("H")-8).":".date("i:s");          //输出时间
            break;                                       //结束语句
            case "日本";                                  //如果是日本
            echo(date("H")+1).":".date("i:s");           //输出时间
            break;                                       //结束语句
            case "开罗";                                  //如果是开罗
            echo(date("H")-6).":".date("i:s");           //输出时间
            break;                                       //结束语句
            case "莫斯科";                                //如果是莫斯科
            echo(date("H")-5).":".date("i:s");           //输出时间
            break;                                       //结束语句
        }
    }
?>
```

技术要点

因为在 PHP 语言中，日期、时间函数依赖于服务器的地区设置，而 PHP 默认设置的



是标准的格林威治时间（即采用的是零时区），所以如果没有对 PHP 的时区进行设置，那么当前使用日期、时间函数获取的将是英国伦敦本地时间（即零时区的时间）。

这也就是我们为什么要对 PHP 的时区进行设置的原因，如果不设置正确的时区，那么 PHP 的日期、时间函数就获取不到正确的当地时间。以东八区为例，我们当地使用的是北京时间，如果没有对 PHP 的时区进行设置，那么获取的时间就将比当地的北京时间少 8 个小时。

更改 PHP 语言中的时区设置有两种方法：

（1）在 php.ini 文件中，定位到[date]下的“;date.timezone =”选项，去掉前面的分号，并设置它的值为当地所在时区使用的时间。修改内容如图 9.2 所示。



图 9.2　设置 PHP 的时区

例如，如果当地所在时区为东八区，那么就可以设置“date.timezone =”的值为：PRC、Asia/Hong_Kong、Asia/Shanghai（上海）或者 Asia/Urumqi（乌鲁木齐）等。这些都是东八区的时间。

设置完成后，保存文件，重新启动 Apache 服务器。

（2）在应用程序中，在日期、时间函数之前使用 date_default_timezone_set()函数就可以完成对时区的设置。date_default_timezone_set()函数的语法如下：

```
date_default_timezone_set(timezone);
```

参数 timezone 为 PHP 可识别的时区名称，如果时区名称 PHP 无法识别，则系统采用 UTC 时区。

例如，设置北京时间可以使用的时区包括：PRC（中华人民共和国）、Asia/Chongqing（重庆），Asia/Shanghai（上海）或者 Asia/Urumqi（乌鲁木齐）。这几个时区名称是等效的。

实例 166　计算考试时间

（实例位置：配套资源\SL\09\166）

实例说明

计算考试时间程序是一种单纯的将两个时间点的时间戳做算术运算。本实例通过 time()函数实现考试时间的计算，运行结果如图 9.3 所示。

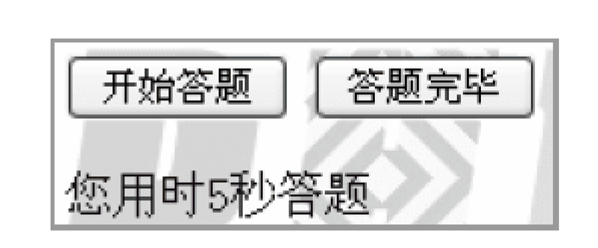
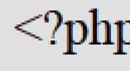


图 9.3　计算考试时间

实现过程

（1）创建 PHP 脚本文件。首先编写<form>表单并设置两个提交按钮，当每个按钮被单击后，利用 SESSION 变量保存此时刻的时间戳，并进行求差运算，然后输出结果。其代码如下：



```
<?php
session_start();
```



```
    header("Content-type:text/html; charset=utf-8");
    echo "<form action ='' method ='post'>";                    //输出表单
    echo "<input type = 'submit' name='sub' value='开始答题'> <input type='submit' name=
'sub_1'value ='答题完毕'>";
    echo "</form>";
    if($_POST['sub']){                                          //通过 POST 方式传参
      $time = time();                                           //取得当前时间戳
      $_SESSION['time'] = $time;                                //将时间戳保存在 SESSION 中
      echo "<script>alert('单击确定开始答题');</script>";          //输出提示
    }
    if($_POST['sub_1']){                                        //通过 POST 方式传参
      if(isset($_SESSION['time'])){                             //判断 SESSION 是否存在
          echo "您用时".(time()-$_SESSION['time'])."秒答题";       //输出结果
          session_destroy();                                    //销毁 SESSION
      }else{                                                    //否则输出提示
          echo "<script>alert('您还没有答题无法结束，请先答题！');location.href='index.php'</script>";
      }
    }
    ?>
```

（2）将文件存储于\MR\09\002\文件夹下，并命名为 index.php。运行结果如图 9.3 所示。

技术要点

本实例的关键点是 time()函数的灵活运用。当单击“开始答题”按钮时，将 time()函数获取的时间戳存储到 SESSION 变量中；当单击“答题完毕”按钮时，再次通过 time()函数获取系统的当前时间戳。通过当前时间戳减去开始定义的时间戳，就是考试所用时间。

实例 167　输出中文格式的日期和时间

（实例位置：配套资源\SL\09\167　视频位置：配套资源\SP\09\167）

实例说明

本实例使用 date()函数实现中文格式的日期和时间的输出，其运行效果如图 9.4 所示。

图 9.4　输出中文格式的日期和时间

实现过程

本实例采用的是 date()函数，定义传递到此函数的参数为“Y 年 m 月 d 日 H 时 i 分 m 秒”。其代码如下：

```
<?php
  header("Content-Type:text/html;charset=utf-8");               //设置编码集
  date_default_timezone_set("Asia/Shanghai");                   //设置时区
  echo date("Y 年 m 月 d 日 H 时 i 分 m 秒");                      //输出信息
?>
```





技术要点

本实例先使用 date_default_timezone_set()函数设置时区，然后使用 date()函数获取时间参数并输出显示。

实例 168　检验日期和时间的有效性

（实例位置：配套资源\SL\09\168）

实例说明

一年有 12 个月、一个月有 31 天（或 30 天，2 月有 28 天，闰年为 29 天），一星期有 7 天……这些都是基本常识。但计算机并不能自己分辨数据的对与错，只是依靠开发者提供的功能去执行或检查。在 PHP 中，通过 checkdate()函数可以检验日期和时间的有效性。运行本实例，效果如图 9.5 所示。

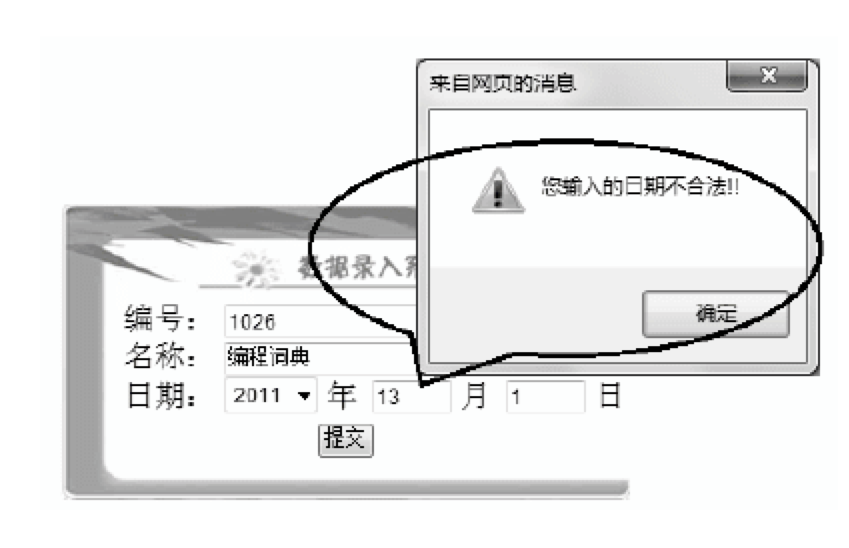


图 9.5　检验日期和时间的有效性

实现过程

具体步骤如下：

（1）创建 index.php 文件，通过 form 表单提交数据信息，以及数据录入时间。

（2）创建 index_ok.php 文件，获取表单提交的数据，完成对提交日期格式的验证。其关键代码如下：

```
<?php
header("Content-type:text/html;charset=utf-8");                    //设置页面编码格式
if(isset($_POST['Submit'])){                                        //获取提交的日期数据
  if(checkdate($_POST['month'],$_POST['day'],$_POST['year'])){     //验证日期格式是否正确
        echo "提交日期合法!!";
  }else{
        echo "<script> alert('您输入的日期不合法!!'); history.back();</script>";
  }
}
?>
```

技术要点

本实例主要使用 checkdate()函数传递三个参数判断某年某月是否存在某日的操作。

checkdate()函数验证日期的有效性，如果日期有效则返回 true，否则返回 false。其语法如下：

```
bool checkdate ( int month, int day, int year)
```

checkdate()函数的参数说明如表 9.1 所示。

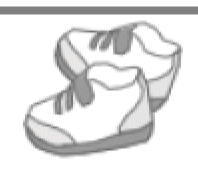




表 9.1　checkdate()函数的参数说明

参　数	说　明
month	month 的有效值是从 1～12
day	day 的有效值在给定的 month 所应该具有的天数范围之内，包括闰年
year	year 的有效值是从 1～32767

实例 169　倒计时

（实例位置：配套资源\SL\09\169　视频位置：配套资源\SP\09\169）

实例说明

倒计时是人们在生活中经常会用到的一个功能，例如，2010 年上海世博会的倒计时、2012 年春节的倒计时等。本实例应用 PHP 的日期、时间函数为 2012 年元旦设计一个倒计时程序，如图 9.6 所示。

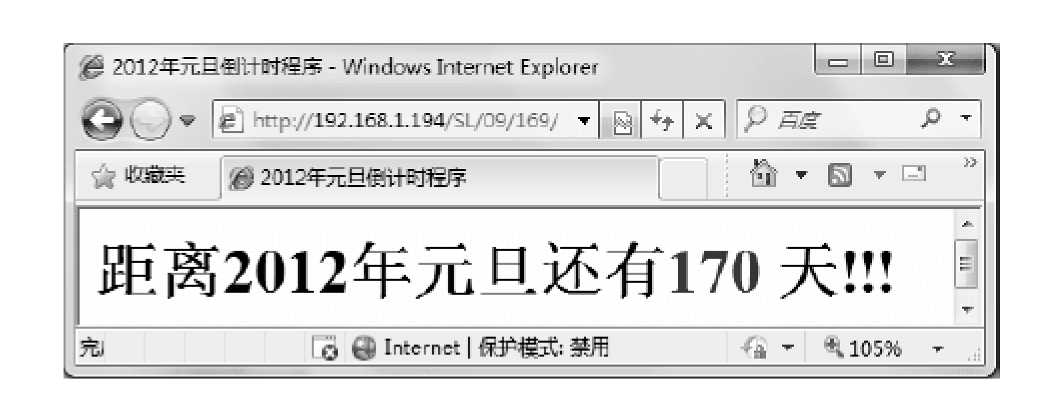


图 9.6　倒计时

实现过程

创建 index.php 文件，编写实现代码，具体内容如下：

```
<?php
	$time1 = strtotime(date( "Y-m-d"));			//当前的系统时间
	$time2 = strtotime("2012-1-1");				//2012 年元旦
	$sub2 = ceil(($time2 - $time1) / 86400);		//（60 秒*60 分*24 小时）秒/天
   echo "距离 2012 年元旦还有<font color=red>$sub2 </font>天!!!";
?>
```

技术要点

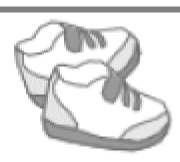
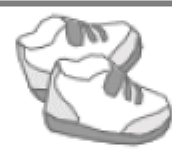
倒计时的原理非常简单，就是用一个固定的时间减去当前的时间，所得到的就是剩余时间。要完成时间的加减操作，同比较大小类似，都要将时间转换成时间戳，然后才能计算，最后再将时间戳转换成日期输出。

实例 170　判断时间的早晚

（实例位置：配套资源\SL\09\170　视频位置：配套资源\SP\09\170）

实例说明

在 PHP 语言中，要完成对两个时间早晚的比较，必须先将时间转换为时间戳，然后才可以进行比较。而将时间转换为时间戳可以通过 strtotime()函数来完成。其运行效果如图 9.7 所示。

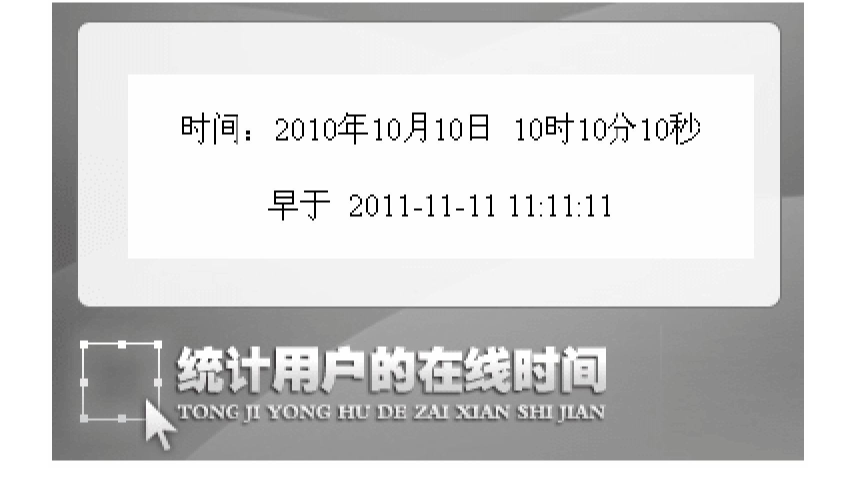


图 9.7　判断时间的早晚

实现过程

首先定义两个固定的时间，然后通过 strtotime()函数将固定时间转换成时间戳，最后对两个时间戳的值进行比较。其代码如下：

```
<?php
  $time = "2010 年 10 月 10 日  10 时 10 分 10 秒";          //设置固定时间
  $times = "2011-11-11 11:11:11";                             //设置固定时间
  if(strtotime($time) - strtotime($times) < 0){               //对两个时间进行运算
        echo "时间：".$time."<p>  早于  ".$times ;             //time－times<0 说明 time 的时间在前
  }else{
        echo "时间：".$times."<p>  早于  ".$time;              //否则，说明 times 的时间在前
  }
?>
```

技术要点

PHP 中应用 strtotime()函数将任何英文文本的日期解析为 UNIX 时间戳，其值为相对于 now 参数给出的时间，如果没有提供此参数，则用系统当前时间，此时与实例 142 中的 time()函数相同。strtotime()函数的语法如下：

```
int strtotime ( string time [, int now] );
```

该函数有两个参数。如果参数 time 的格式是绝对时间，则 now 参数不起作用；如果参数 time 的格式是相对时间，其对应的时间就是参数 now 来提供的，当没有提供参数 now 时，对应的时间就为当前时间。如果解析失败，则返回 false。在 PHP5.1.0 之前的版本中，本函数在失败时返回-1。

实例 171　网页闹钟

（实例位置：配套资源\SL\09\171）

实例说明

闹钟是生活中经常使用的一个小工具，它会在指定时间叫你起床。在 Web 程序中也可以应用这个原理，开发一个“网页闹钟”，提示用户在指定的时间或者时间段内要做什么工作。下面编写一个实例，当系统的当前时间运行到 9 月 18 日，给出提示信息“勿忘国耻！”，其运行效果如图 9.8 所示。

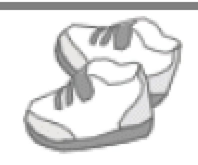




图 9.8　网页闹钟



实现过程

所谓网页闹钟也就是一个日志提醒功能，通过判断系统中当前的时间与指定的某个时间或者时间段是否相同，如果相同，系统就给出一个对应的提示信息，提示用户该做什么（例如，每天工作记录的上传、每月经验技巧的提交或者员工生日的提醒等），都可以通过日志的形式进行提醒。在本实例中，通过当前时间戳与 9 月 18 日的时间戳进行比较，如果相同，则给出提示信息。其关键代码如下：

```
<?php
$time1 = strtotime(date("Y-m-d"));                    //当前的系统时间，获取月和天的时间戳
$time2 = strtotime(date("Y")."-09-18");               //设置时间 9 月 18 日的时间戳
if($time1==$time2){                                   //判断两个时间戳是否相同
   echo "<script>alert('勿忘国耻！');window.location.href='index.php';</script>"; //给出提示信息
}else{
  echo "今天不是一个特殊的日子！";
}
?>
```

技术要点

本实例主要使用 strtotime()函数，PHP 中应用 strtotime()函数将任何英文文本的日期解析为 UNIX 时间戳，其值为相对于 now 参数给出的时间，如果没有提供此参数则用系统当前时间。strtotime 函数的语法如下：

```
int strtotime ( string time [, int now] );
```

该函数有两个参数。如果参数 time 的格式是绝对时间，则 now 参数不起作用；如果参数 time 的格式是相对时间，其对应的时间就是参数 now 来提供的，当没有提供参数 now 时，对应的时间就为当前时间。如果解析失败，则返回 false。在 PHP5.1.0 之前的版本中，本函数在失败时返回-1。

实例 172　计算程序的运行时间

（实例位置：配套资源\SL\09\172　视频位置：配套资源\SP\09\172）

实例说明

在百度中，当我们执行一个查询操作时，在获取到查询结果后，页面中就会出现一行文字，提示根据关键字搜索到多少个结果，以及搜索所用的时间。这个时间就是程序在执行这个搜索时所用的时间，那么它是如何实现的呢？本实例模仿它来计算程序运行的时





间。运行结果如图 9.9 所示。

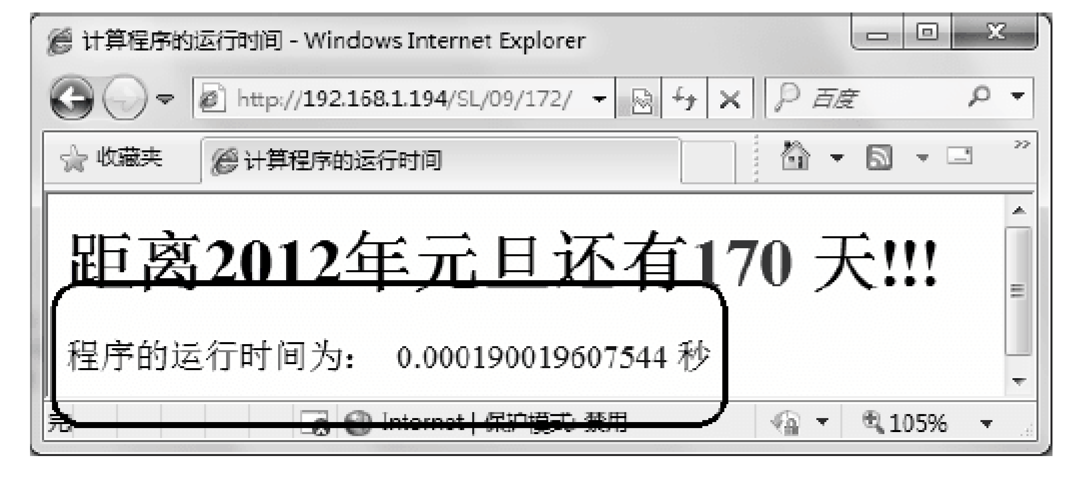


图 9.9 计算程序的运行时间

实现过程

首先，在执行查询操作之前定义一个时间，将其精确到微秒。然后，执行查询操作，并在查询功能执行完毕后再次获取一个时间，同样精确到微秒。最后，应用后获取的时间减去最初获取的时间，就是本次查询所用的时间。其关键代码如下：

```
<?php
/*      声明 run_time 函数*/
function run_time(){
list($msec, $sec) = explode(" ", microtime());          //使用 explode 函数返回两个变量
return ((float)$msec + (float)$sec);                    //返回两个变量的和
}
$start_time = run_time();                               //第一次运行 run_time()函数
/*      运行 PHP 代码段      */
$end_time = run_time();                                 //再次运行 run_time()函数
?>
```

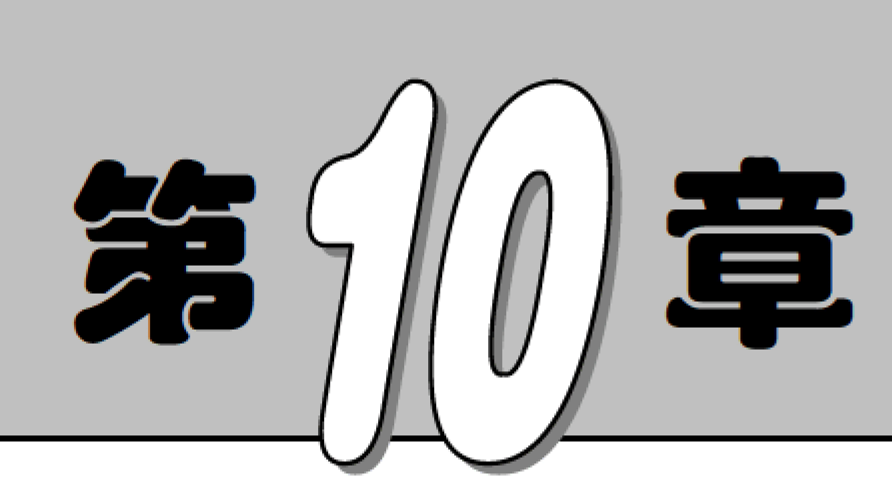
技术要点

本实例主要通过 microtime()获取当前时间的微秒数和时间戳。

microtime()函数，返回当前 UNIX 时间戳和微秒数，返回格式为“msec sec”的字符串，其中 sec 是当前的 UNIX 时间戳，msec 是微秒数。本函数仅在支持 gettimeofday()函数的操作系统下可用，其语法如下：

```
string microtime ( void )
```

应用 explode()函数将 microtime()函数返回的字符串进行分隔，返回一个数组，包括两个元素：一个元素是当前时间的微秒数，另一个是当前时间的时间戳。应用 list()函数将 explode()函数返回的数组值赋给指定的变量。最后，获取两个变量值的和。

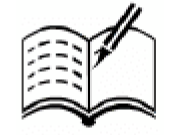


图形图像处理

本章读者可以学到如下实例：

- 实例 173　GD2 函数填充几何图形
- 实例 174　GD2 函数在照片上添加文字
- 实例 175　GD2 函数为图片添加文字水印
- 实例 176　GD2 函数为图片添加图像水印
- 实例 177　GD2 函数生成图形验证码
- 实例 178　折线图分析 2010 年的销售额
- 实例 179　柱形图分析编程词典销售比例
- 实例 180　饼形图分析 2010 年图书销量
- 实例 181　GD2 函数折线图分析网站月访问量走势
- 实例 182　GD2 函数柱状图分析编程词典满意度调查
- 实例 183　GD2 函数饼形图分析图书市场的份额
- 实例 184　柱状图展示编程词典 6、7 月份销售量
- 实例 185　柱状图展示编程词典上半年销量
- 实例 186　折线图分析 2010 年牛肉市场价格走势
- 实例 187　多饼形图区块分析 2010 年图书销量
- 实例 188　缩略图艺术库
- 实例 189　通过图像显示投票统计结果
- 实例 190　通过图像显示密码安全强度
- 实例 191　任意调整上传图片的大小





实例 173　GD2 函数填充几何图形

（实例位置：配套资源\SL\10\173　视频位置：配套资源\SP\10\173）

实例说明

使用 GD2 函数不仅可以绘制线条图形，而且可以绘制填充图形，如填充圆形、填充矩形等。在本实例中，将介绍圆形和矩形的填充，其运行结果如图 10.1 所示。



图 10.1　GD2 函数填充几何图形

实现过程

本实例应用 GD2 函数绘制填充圆形和填充矩形，其代码如下：

```
<?php
header("Content-type: image/png");                          //将图像输出到浏览器
$img = imagecreate(400, 200);                               //创建一个 400×200 的图像
$bg = imagecolorallocate($img, 0, 0, 255);
$white = imagecolorallocate($img, 255, 0, 255);
imagefilledellipse($img, 100, 100, 150, 150, $white);       //绘制圆形
imagefilledrectangle($img, 200, 50, 300, 150, $white);      //绘制矩形
imagepng($img);
imagedestroy($img);
?>
```

技术要点

在 GD2 函数库中，应用 imagefill()函数实现图像的填充操作，其语法如下：

```
bool imagefill ( resource image, int x, int y, int color)
```

imagefill()函数在 image 图像的坐标 x, y（图像左上角为 0,0）处用 color 颜色执行区域填充（即与 x, y 点颜色相同且相邻的点都会被填充）。

imagefilledarc()函数绘制一个椭圆弧线，其语法如下：

```
bool imagefilledarc ( resource image, int cx, int cy, int w, int h, int s, int e, int color, int style )
```

imagefilledarc()函数在 image 所代表的图像中以 cx，cy（图像左上角为 0,0）为坐标点绘制一椭圆弧。如果成功，则返回 true；否则，返回 false。w 和 h 分别指定椭圆的宽和高，s 和 e 参数以角度指定起始和结束点。style 可以是下列值按位或（OR）后的值：

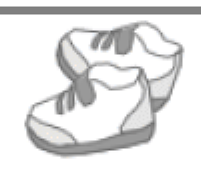
（1）IMG_ARC_PIE。

（2）IMG_ARC_CHORD。

（3）IMG_ARC_NOFILL。

（3）IMG_ARC_EDGED。

IMG_ARC_PIE 和 IMG_ARC_CHORD 是互斥的。IMG_ARC_CHORD 只是用直线连




Note

接起始和结束点，IMG_ARC_PIE 则产生圆形边界（如果两个都用，IMG_ARC_CHORD 生效）。IMG_ARC_NOFILL 指明弧或弦只有轮廓，不填充。IMG_ARC_EDGED 指明用直线将起始和结束点与中心点相连，和 IMG_ARC_NOFILL 一起使用是绘制饼状图轮廓的好方法（而不用填充）。

Imagefilledellipse()函数绘制一个椭圆，其语法如下：

```
bool imagefilledellipse ( resource image, int cx, int cy, int w, int h, int color )
```

该函数在 image 所代表的图像中以 cx, cy（图像左上角为 0,0）为中心绘制一个椭圆。w 和 h 分别指定椭圆的宽和高，使用 color 颜色填充。如果成功则返回 true，否则返回 false。

指点迷津：

在通过 GD2 函数库创建图像时，imagedestroy()函数是必不可少的，通过它可以销毁指定的图像，释放与 image 关联的内存。如果不应用该函数，那么与 image 关联的数据会一直存储在内存中。

实例 174　GD2 函数在照片上添加文字

（实例位置：配套资源\SL\10\174　视频位置：配套资源\SP\10\174）

实例说明

PHP 中的 GD 库支持中文，但必须要以 UTF-8 格式的参数来进行传递。如果使用 imageString()函数直接绘制中文字符串，就会显示乱码，这是因为 GD2 对中文只能接收 UTF-8 编码格式，并且默认使用了英文的字体，所以要输出中文字符串，就必须对中文字符串进行转码，并设置中文字符使用的字体；否则，输出的只能是乱码。在本实例中，实现 GD2 函数输出中文字符串，并且通过 GD2 函数将中文字符串在照片上输出，其运行结果如图 10.2 所示。

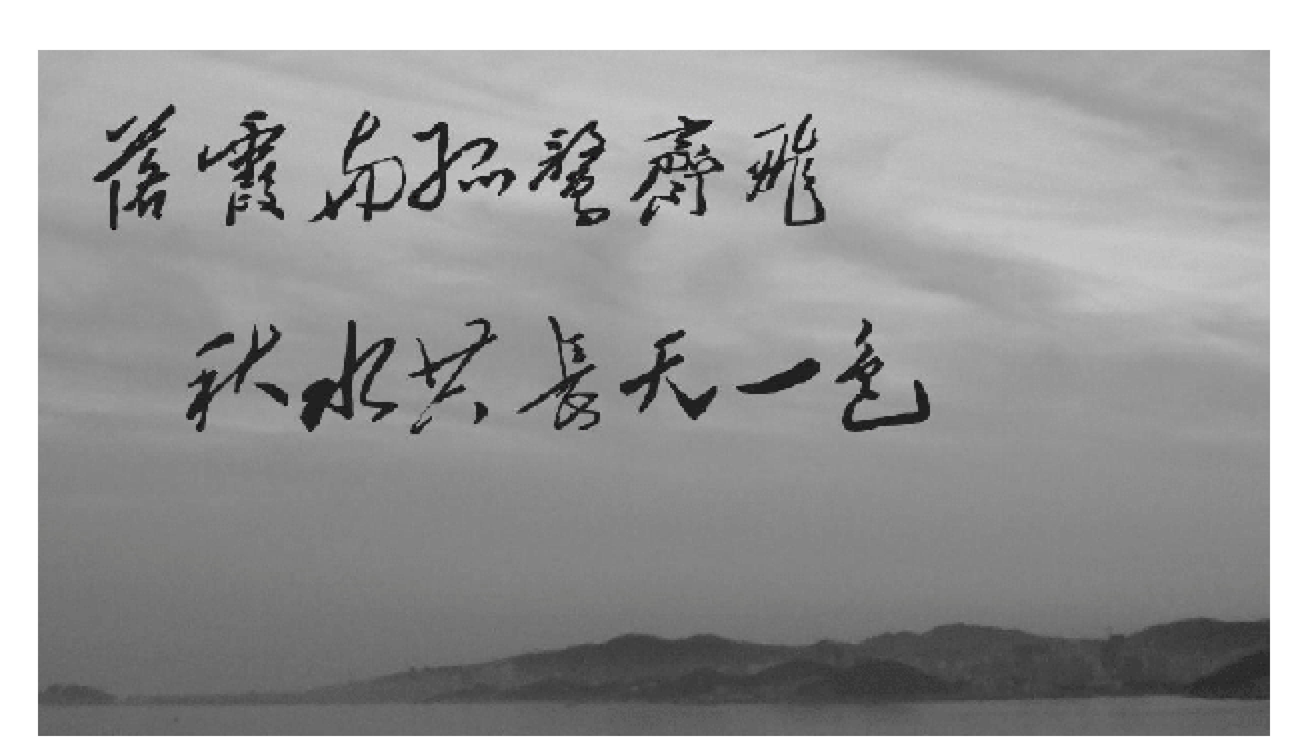
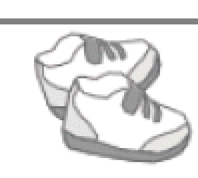


图 10.2　在照片上添加文字

实现过程

本实例应用 imagecreatefromjpeg()和 imagettftext()函数在照片上添加中文字符串。其代





码如下：

```
<?php
header("content-type:image/jpeg");                    //定义输出为图像类型
$im=imagecreatefromjpeg("images/P1020494.JPG");       //载入照片
$textcolor=imagecolorallocate($im,25,25,136);      //设置字体颜色为蓝色，值为 RGB 颜色值
$font="Font/mzd.ttf";                                 //定义字体
$to="落霞与孤鹜齐飞";
imagettftext($im,80,0,200,200,$textcolor,$font,$to);  //写 TTF 文字到图中
$to="秋水共长天一色";
imagettftext($im,80,0,300,400,$textcolor,$font,$to);  //写 TTF 文字到图中
imagejpeg($im);                                       //建立 JPEG 图形
imagedestroy($im);                                    //结束图形，释放内存空间
?>
```

多学两招：

因为本实例中页面使用的是 UTF-8 编码格式，所以在定义向图片中添加的中文字符串时可以直接使用，不需要编码格式的转换。但是，如果页面本身使用的是 GB2312 编码格式，那么就必须应用 iconv()函数对向图片中添加的中文字符串进行编码格式的转换，由 GB2312 编码转换为 UTF-8 编码。

技术要点

在 PHP 中向图像中添加中文字符串应用的是 imagettftext()函数，其语法如下：

```
array imagettftext ( resource image, float size, float angle, int x, int y, int color, string fontfile, string text )
```

imagettftext ()函数的参数说明如表 10.1 所示。

表 10.1　imagettftext()函数的参数说明

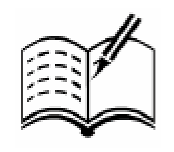
参　　数	说　　明
image	图像资源
size	字体大小。根据 GD 版本不同，应该以像素大小指定（GD1）或点大小（GD2）
angle	字体的角度，顺时针计算，0° 为水平，也就是 3 点钟的方向（由左到右），90° 则为由下到上的文字
x	文字的 x 坐标值。设定第一个字符的基本点
y	文字的 y 坐标值。设定字体基线的位置，不是字符的最底端
color	文字的颜色
fontfile	字体的文件名称，也可以是远端的文件
text	字符串内容

在 PHP 中，通过 GD2 函数对 JPG 格式的照片进行操作，应用的是 imagecreatefromjpeg()函数，从 JPEG 文件或 URL 新建一图像。其语法如下：

```
resource imagecreatefromjpeg ( string filename )
```

参数 filename 可以是本地文件，也可以是网络的 URL 地址。返回值为 JPEG 的文件代





码，可供其他函数使用。该函数在失败时返回一个空字符串，并且输出一条错误信息。

实例 175　GD2 函数为图片添加文字水印

（实例位置：配套资源\SL\10\175）

实例说明

图片是 Web 页面最为重要的组成元素之一，新闻网站、图片资料网站等备受网民关注的网站每天都会上传大量的图片。如果直接将图片上传到页面中，很可能被浏览者保存使用，这样站点的版权就不能得到很好的保证。如果在上传图片过程中，动态地为图片添加水印效果，这样不仅可保护版权，如果设计合理还能有助于网站的推广。在本实例中，将讲解如何在上传图片的过程中为图片添加水印文字，其运行结果如图 10.3 所示。



图 10.3　为图片添加文字水印

实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，创建一个表单，完成上传图片的提交操作。然后，在本页中输出存储在服务器指定文件夹下的图片。最后，获取表单中提交的图片数据，应用 move_uploaded_file()函数完成图片的上传操作，并且实例化存储在 AddWaterPress.php 文件中的 AddWaterPress 类，调用 add()方法为指定的图片添加文字水印。其关键代码如下：

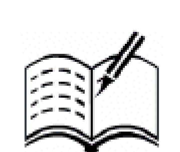
```
if(move_uploaded_file($_FILES["file"]["tmp_name"], $saveDir)){      //执行文件上传操作
require_once 'AddWaterPress.php';                                    //包含添加水印操作的文件
        $addWaterPress = new AddWaterPress();                        //类的实例化
      $addWaterPress->add($saveDir, "吉林省明日科技");//执行添加方法，传递参数，指定水印文字
        echo "<script>alert('图片添加成功');</script>";
   }
```

（2）创建 AddWaterPress.php 文件，编写 AddWaterPress 类的内容。其关键代码如下：

```
<?php
class AddWaterPress{                                                  //定义类文件
//省略了部分代码
  function add($imageUrl, $watherImageUrl){                           //定义添加方法
  $img = @$this->getImageRes($this->getExtendsName($imageUrl), $imageUrl); //获取被操作的图
像标识
        $textcolor=imagecolorallocate($img,190,1,23);                 //设置字体颜色为蓝色，
值为 RGB 颜色值
        $font="Font/FZHCJW.TTF";                                      //定义字体
        imagettftext($img,15,56,20,130,$textcolor,$font,$watherImageUrl);  //写 TTF 文字到图中
```



```
            //根据图像标识符、后缀和路径，执行 outputImage 方法，输出图像
            $this->outputImage($img, $this->getExtendsName($imageUrl), $imageUrl);
          imagedestroy($img);                                        //销毁图像
      }
  }
```



Note

多学两招：

本实例完成的是中文字符串水印标记的添加操作，如果要添加英文字符串格式的水印标记，那么就可以直接使用 imageString()函数来完成。其方法与在图像中输出英文字符串是相同的。

技术要点

上传图片添加水印的重点是图像处理函数的运用，至于文件的上传已经在前面的实例中进行了详细的讲解，这里就不再赘述。

（1）创建图像函数。

创建图像函数的运用是完成这个实例的第一步，因为需要根据上传图片的后缀（格式）运用不同的函数来创建新图像，这里定义 3 种格式的图片（gif、jpg 和 png），所以要应用到 3 个创建图像函数。

☑ imagecreatefromgif()：根据 GIF 文件或者 URL 新建一图像，返回图像标识符。
☑ imagecreatefromjpeg()：根据 JPG 文件或者 URL 新建一图像，返回图像标识符。
☑ imagecreatefrompng()：根据 PNG 文件或者 URL 新建一图像，返回图像标识符。

（2）操作图像函数。

新图像创建成功后将返回一个图像标识符，操作图像函数根据标识符对图像进行操作，这里应用 2 个函数：

☑ imagecolorallocate()：定义水印文字的颜色。
☑ imagettftext()：在返回的图像中输入中文字符串。

（3）输出图像函数。

与创建图像函数对应，同样有 3 个输出图像函数。

☑ imagegif()：以 GIF 格式将图像输出到浏览器或文件。
☑ imagejpeg()：以 JPG 格式将图像输出到浏览器或文件。
☑ imagepng()：以 PNG 格式将图像输出到浏览器或文件。

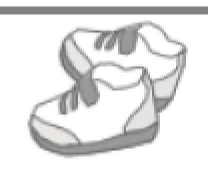
（4）销毁图像函数。

在完成图像的创建和输出之后，有必要销毁图像，释放与 image 相关的内存。此时，应用的是 imagedestroy()函数。其语法如下：

```
bool imagedestroy ( resource image )
```

imagedestroy()函数释放与 image 关联的内存。参数 image 是由图像创建函数返回的图像标识符。

在本实例中，对上述介绍的函数进行整合应用，创建 AddWaterPress 类，定义 getExtendsName()方法，获取上传图片的文件后缀；定义 getImageRes()方法，根据上传文





件的后缀创建新图像；定义 outputImage()方法，输出图像；定义 add()方法，向指定的图片中添加文字水印。

实例 176　GD2 函数为图片添加图像水印

（实例位置：配套资源\SL\10\176）

实例说明

实例 175 介绍的是如何为照片添加文字水印，在本实例中将介绍如何为上传图片添加一个图像水印，运行结果如图 10.4 所示。



图 10.4　为图片添加图像水印

实现过程

本实例的实现步骤与实例 175 相同，唯一区别是创建 AddWaterPress.php 文件，编写 AddWaterPress 类的 add()方法时，应用的是 getimagesize()和 imagecopy()函数完成图像水印的添加操作。Add()方法的代码如下：

```
function add($imageUrl, $watherImageUrl, $x, $y){          //定义添加方法
    $img = @$this->getImageRes($this->getExtendsName($imageUrl), $imageUrl); //获取被添加的图像标识
    $img1 = @$this->getImageRes($this->getExtendsName($watherImageUrl), $watherImageUrl);
    $size = getimagesize($imageUrl);                  //获取图像大小
    $size1 = getimagesize($watherImageUrl);           //获取水印图片的大小
    if($x==null && $y==null){                         //判断参数是否为空
        $x1 = ($size[0]-$size1[0])/2;//根据图像大小数组中返回的值，计算图像的横坐标
        $y1 = ($size[1]-$size1[1])/2;//根据图像大小数组中返回的值，计算图像的纵坐标
    }else{
        $x1 = $x;                                     //如果不为空，则直接使用坐标数据
        $y1 = $y;                                     //如果不为空，则直接使用坐标数据
    }
    imagecopy($img, $img1, $x1, $y1, 0, 0, $size1[0], $size1[1]);//将 img1 的一部分复制到 img 的指定位置
    //根据图像标识符、后缀和路径，执行 outputImage 方法，输出图像
    $this->outputImage($img, $this->getExtendsName($imageUrl), $imageUrl);
    imagedestroy($img1);                              //销毁图像
    imagedestroy($img);                               //销毁图像
}
```

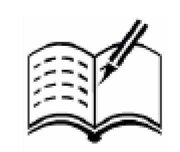
技术要点

使用图片来作为水印，前提是这个图片的背景必须是透明的，否则输出的效果很不理



想。图片水印添加的关键是 getimagesize()和 imagecopy()函数。应用 getimagesize()函数获取上传图片和水印图片的大小，通过 imagecopy()函数完成图片水印的添加。

imagecopy()函数，将图像复制到指定的另外一个图像中。其语法如下：



```
bool imagecopy ( resource dst_im, resource src_im, int dst_x, int dst_y, int src_x, int src_y, int src_w, int src_h )
```

将 src_im 图像中坐标从 src_x，src_y 开始，宽度为 src_w，高度为 src_h 的一部分复制到 dst_im 图像中坐标为 dst_x 和 dst_y 的位置上。

在本实例中，同样是整合图像的创建、操作、输出和销毁函数，创建 AddWaterPress 类，定义 getExtendsName()方法，获取上传图片的文件后缀；定义 getImageRes()方法，根据上传文件的后缀创建新图像；定义 outputImage()方法，输出图像；定义 add()方法，向指定的图片中添加图像水印。

实例 177 GD2 函数生成图形验证码

（实例位置：配套资源\SL\10\177 视频位置：配套资源\SP\10\177）

实例说明

验证码技术的应用是为了提高站点的安全性，避免因网页运行速度慢而造成数据的重复提交。本实例将通过 JavaScript 脚本和 GD2 函数开发一个无刷新验证码，运行结果如图 10.5 所示。

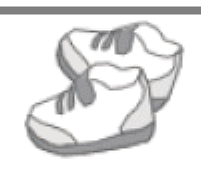


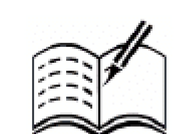
图 10.5 GD2 函数生成图形验证码

实现过程

具体步骤如下：

（1）创建 index.php 页面，定义 form 表单，设计用户注册页面的效果。





（2）创建 JavaScript 脚本，生成验证码；并定义 reCode()方法，用于验证码的二次生成。其关键代码如下：

```
<script language="javascript">
var num1=Math.round(Math.random()*10000000);              //生成随机数
var num=num1.toString().substr(0,4);                      //截取随机数的前 4 个字符
document.write("<img name=codeimg src='ValidatorCode.php?code="+num+"'>"); //将值传递到图像页
form1.defValidatorCode.value=num;                         //将截取值赋给表单中的隐藏域
</script>
```

（3）创建超链接，调用 JavaScript 脚本中的 reCode()方法，实现验证码的二次生成。其代码如下：

```
<a href="javascript:reCode()" class="a1">看不清</a>
```

（4）创建 ValidatorCode.php 页面，根据超链接中传递的随机数，应用 GD2 函数生成验证码的数字图像。其关键代码如下：

```
<?php
header('content-type:image/png');                                    //定义标题 png 格式图像
$im = imagecreate(65, 25);                                           //定义画布
imagefill($im, 0, 0, imagecolorallocate($im, 200, 200, 200));        //区域填充
$validatorCode = $_GET['code'];                                      //获取提交的值
imagestring($im, rand(3, 5), 10, 3, substr($validatorCode, 0, 1), imagecolorallocate($im, 0, rand(0, 255),
rand(0, 255)));
imagestring($im, rand(3, 5), 25, 6, substr($validatorCode, 1, 1), imagecolorallocate($im, rand(0, 255), 0,
rand(0, 255)));
imagestring($im, rand(3, 5), 36, 9, substr($validatorCode, 2, 1), imagecolorallocate($im, rand(0, 255), rand(0,
255), 0));
imagestring($im, rand(3, 5), 48, 12, substr($validatorCode, 3, 1), imagecolorallocate($im, 0, rand(0, 255),
rand(0, 255)));
imagepng($im);                                                       //生成 png 图像
imagedestroy();                                                      //销毁图像
```

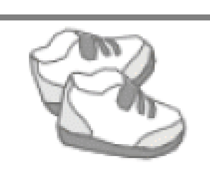
（5）在 js 文件夹下，创建一个 js 脚本文件，定义 chkinput()方法，对表单中提交的注册信息进行验证，包括验证码是否正确。其完整代码请参考本书配套资源。

技术要点

应用 JavaScript 脚本和 GD2 函数开发一个无刷新验证码涉及的技术包括：随机数的生成，随机数如何传递给图像生成页，数字图像的生成，以及验证码的二次生成。这里重点介绍验证码图像的生成。

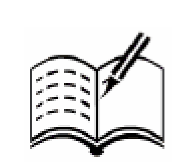
（1）创建 ValidatorCode.php，应用 GD2 函数库中的函数，根据超链接传递的参数值生成验证码图像。

- ☑ imagecreate()函数，创建验证码图像的画布。
- ☑ imagecolorallocate()函数，定义画布的填充颜色。
- ☑ imagefill()函数，根据 imagecolorallocate()函数定义的颜色，完成图像颜色的填充



操作。

☑ imagestring()函数，将从超链接中获取的验证码值写入到图像中，并且应用 imagecolorallocate()函数随机定义值输出的颜色。



☑ imagepng()函数，生成 PNG 格式的图像。

☑ imagedestroy()函数，销毁图像，释放内存。

（2）验证码的二次生成。其实现原理类似于 do...while 循环语句，无论条件是否满足，它默认都会执行一次（打开网页后，直接生成一个验证码）。然而，要想其继续执行，就必须满足指定的条件（重新生成验证码，就必须调用指定的方法）。依据这个原理，在 JavaScript 脚本中定义如下方法：

```
function reCode(){                                              //定义方法，重新生成验证码
  var num1=Math.round(Math.random()*10000000);                  //生成随机数
  var num=num1.toString().substr(0,4);                          //截取随机数
  document.codeimg.src="ValidatorCode.php?code="+num;           //将截取值传递到图像处理页中
  form1.defValidatorCode.value=num;                             //将截取值赋给表单中的隐藏域
}
```

在上述的方法中，重复执行 JavaScript 脚本中默认的操作，即重新生成一个验证码。

多学两招：

在本实例中，应用 GD2 函数生成的是清晰的数字图形验证码，还可以在此基础上为验证码增加干扰背景，使其看上去模糊一些，如图 10.6 所示。

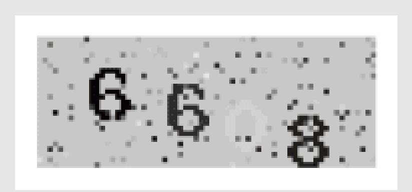


图 10.6　增加干扰背景后的验证码图像

其具体方法如下：在 ValidatorCode.php 文件中，通过 for 循环语句，应用 imagesetpixel()函数在画布的背景上绘制一些单一元素。代码如下：

```
for ($i = 0; $i < 200; $i ++) {                         //填充干扰背景
    imagesetpixel($im, rand() % 70, rand() % 30, imagecolorallocate($im, rand(0, 255), rand(0, 255), rand(0, 255)));
}
```

实例 178　折线图分析 2010 年的销售额

（实例位置：配套资源\SL\10\178　视频位置：配套资源\SP\10\178）

实例说明

本实例运用 Jpgraph 生成折线图，对公司 2010 年的销售额进行分析，运行结果如图 10.7 所示。





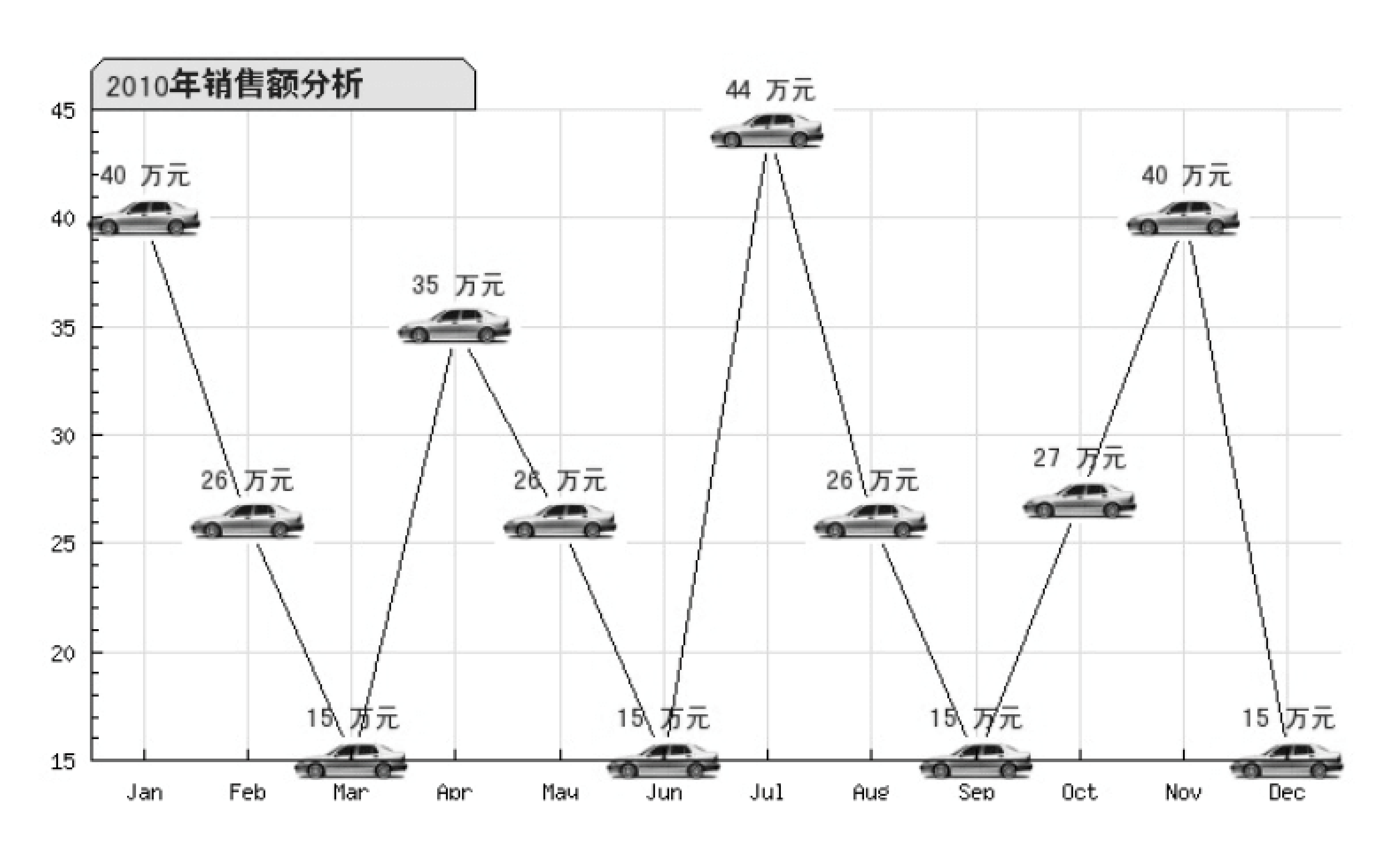


图 10.7　折线图分析 2010 年的销售额

实现过程

具体步骤如下：

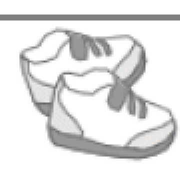
（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

（2）应用 Jpgraph 类库中的方法创建一个折线图，对 2010 年公司的销售额进行分析。其代码如下：

```
$datay1 = array(40,26,15,35,26,15,44,26,15,27,40,15);
$graph = new Graph(650,375);                                            //定义图像大小
$graph->SetMarginColor('white');                                        //背景颜色
$graph->SetScale("textlin");                                            //定义刻度值类型
$graph->SetFrame(false);
$graph->SetMargin(30,5,35,20);                                          //设置边距
$graph->tabtitle->Set(iconv("utf-8","gb2312",'2010 年销售额分析') ); //输出标题
$graph->tabtitle->SetFont(FF_SIMSUN, FS_BOLD,12);                       //设置标题字体
$graph->tabtitle->SetColor('darkred','#E1E1FF');                        //设置标题颜色
$graph->xgrid->Show();                                                  //设置阴影
$graph->xaxis->SetTickLabels($gDateLocale->GetShortMonth());            //定义 X 轴上的数据
$p1 = new LinePlot($datay1);                                            //创建图像
$p1->SetColor("navy");                                                  //设置图像颜色
$p1->mark->SetType(MARK_IMG,'saab_95.jpg',0.5);                         //设置标签的样式，使用图片
$p1->value->SetFormat('%d '.iconv("utf-8","gb2312",'万元'));            //格式化数据
$p1->value->Show();                                                     //输出阴影
$p1->value->SetColor('darkred');                                        //定义颜色
$p1->value->SetFont(FF_SIMSUN, FS_BOLD,10);                             //设置字体
$p1->value->SetMargin(14);                                              //设置位置、字体大小
$p1->SetCenter();
$graph->Add($p1);                                                       //添加数据
$graph->Stroke();                                                       //生成图像
```

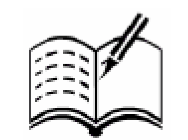
技术要点

本实例通过 Jpgraph 类库创建折线图，分析 2010 年公司的销售额，旨在突出如何在折



线图中通过插入的图像设置数据点标签。

应用图像中 mark 对象的 SetType()方法，实例 177 中运行本方法的第一个参数，直接设置标签的样式，在本实例中运用 SetType()方法的第二个参数，通过插入的图像设置数据点的标签，并通过第三个参数设置插入图像的大小。



Note

实例 179　柱形图分析编程词典销售比例

（实例位置：配套资源\SL\10\179　视频位置：配套资源\SP\10\179）

实例说明

本实例运用 Jpgraph 生成柱形图，对编程词典的销售比例进行分析，运行结果如图 10.8 所示。

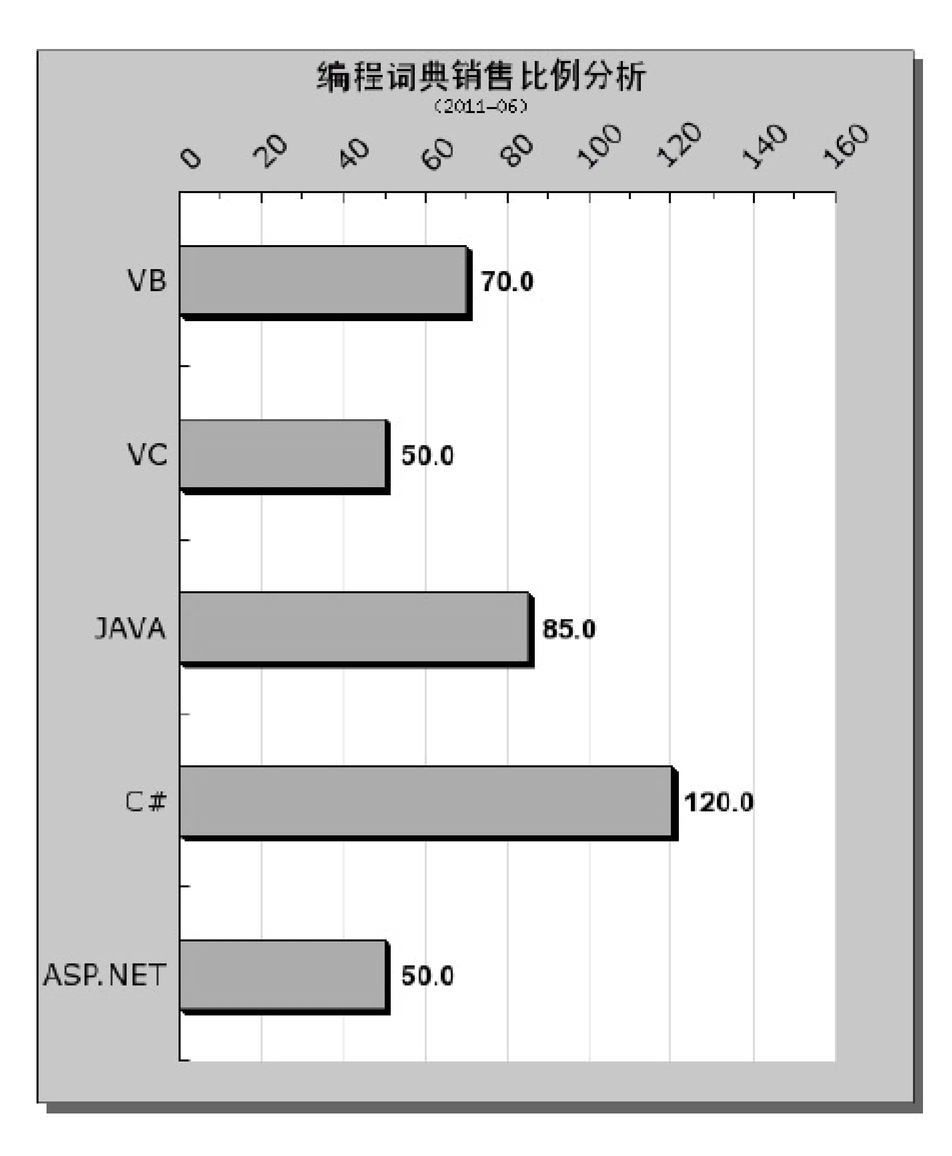


图 10.8　柱形图分析编程词典销售比例

实现过程

具体步骤如下：

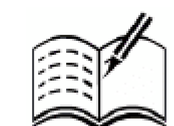
（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

（2）应用 Jpgraph 类库中的方法创建一个柱形图，从另外一个角度分析编程词典的销售比例。其代码如下：

```
$datay=array(70,50,85,120,50);
$datax=array("VB","VC","JAVA","C#","ASP.NET");
$width=500;                                          //定义图像宽
$height=600;                                         //定义图像高
$graph = new Graph($width,$height,'auto');           //创建图像
$graph->SetScale("textlin");                         //定义刻度值类型
$top = 80;
```



```
$bottom = 30;
$left = 80;
$right = 50;
$graph->Set90AndMargin($left,$right,$top,$bottom);                //设置图像的边距
$graph->title->Set(iconv("utf-8","gb2312","编程词典销售比例分析")); //定义标题
$graph->title->SetFont(FF_SIMSUN, FS_BOLD,14);                     //设置标题字体
$graph->subtitle->Set("(2011-06)");                               //设置附标题
$graph->xaxis->SetTickLabels($datax);                             //定义 X 轴上的数据
$graph->xaxis->SetFont(FF_VERDANA,FS_NORMAL,12);                  //设置字体
$graph->xaxis->SetLabelAlign('right','center');                   //设置 Y 轴上字体右侧居中
$graph->yaxis->scale->SetGrace(20);                               //设置 Y 轴间距
$graph->yaxis->SetLabelAlign('center','bottom');               //设置 X 轴字体居中，底部对齐
$graph->yaxis->SetLabelAngle(45);                                 //设置字体的倾斜度
$graph->yaxis->SetLabelFormat('%d');                              //输出数据的格式
$graph->yaxis->SetFont(FF_VERDANA,FS_NORMAL,12);                  //设置字体
$bplot = new BarPlot($datay);                                     //创建图像
$bplot->SetFillColor("orange");                                   //定义图像颜色
$bplot->SetShadow();                                              //设置图像阴影
$bplot->value->Show();                                            //设置数据阴影
$bplot->value->SetFont(FF_ARIAL,FS_BOLD,12);                      //设置数据字体
$bplot->value->SetAlign('left','center');                         //设置数据位置
$bplot->value->SetColor("black","darkred");                       //设置数据颜色
$bplot->value->SetFormat('%.1f');                                 //格式化数据
$graph->Add($bplot);                                              //添加数据
$graph->Stroke();                                                 //生成图像
```



技术要点

本实例通过 Jpgraph 类库创建柱形图，分析编程词典的销售比例。本实例中创建的柱形图与以往的有些不同，这里将柱形图顺时针旋转了 90°。

其主要应用图像对象中的 Set90AndMargin()方法，完成柱形图的顺时针 90°旋转。其语法如下：

```
Set90AndMargin($lm, $rm, $tm, $bm)
```

参数说明：

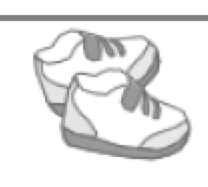
☑ $lm：左边框旋转角度。
☑ $rm：右边框旋转角度。
☑ $tm：上边框旋转角度。
☑ $bm：下边框旋转角度。

实例 180　饼形图分析 2010 年图书销量

（实例位置：配套资源\SL\10\180　视频位置：配套资源\SP\10\180）

实例说明

本实例运用 Jpgraph 生成饼形图，对公司年度总的收支情况进行分析，运行结果如



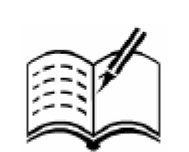


图 10.9 所示。

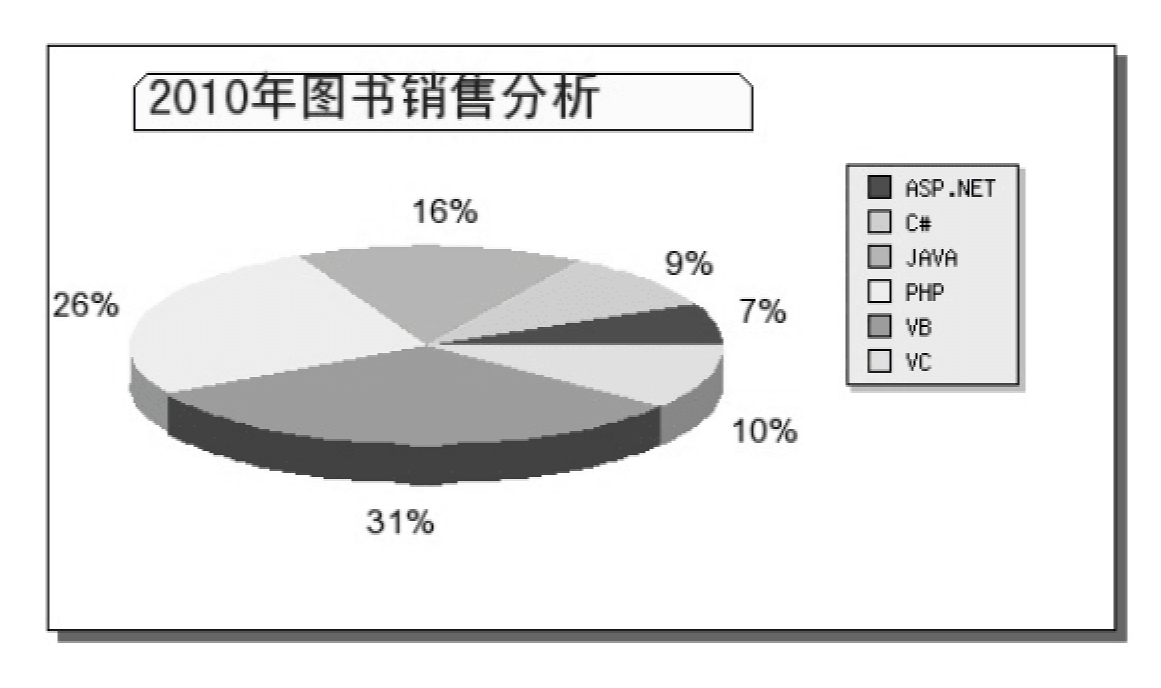


图 10.9　饼形图分析 2010 年图书销量

实现过程

具体步骤如下：

（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

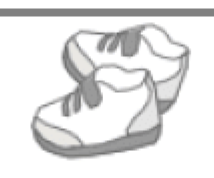
（2）应用 Jpgraph 类库中的方法创建一个饼形图，对 2010 图书销量进行分析。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
include ("../src/jpgraph.php");
include ("../src/jpgraph_pie.php");
include ("../src/jpgraph_pie3d.php");
$data = array(20,27,45,75,90,30);
$graph = new PieGraph(500,275,"auto");                              //创建图像
$graph->SetShadow();                                                //输出阴影
$graph->tabtitle->Set(iconv("utf-8","gb2312",'2010 年图书销售分析') ); //输出标题
$graph->tabtitle->SetFont(FF_SIMSUN, FS_BOLD,18);                   //设置标题字体
$graph->title->SetColor("darkblue");                                //定义标题颜色
$graph->legend->Pos(0.1,0.2);                                       //控制注释文字的位置
$p1 = new PiePlot3d($data);                                         //创建 3D 饼形图图像
$p1->SetTheme("water");                                             //控制图像颜色的主题
$p1->SetCenter(0.35);                                               //控制图像的大小比例
$p1->SetAngle(30);                                                  //控制图像的倾斜角度
$p1->value->SetFont(FF_ARIAL,FS_NORMAL,12);                         //设置字体
$p1->SetLegends(array("ASP.NET","C#","JAVA","PHP","VB","VC"));
$graph->Add($p1);                                                   //添加数据
$graph->Stroke();                                                   //生成图像
?>
```

技术要点

本实例通过 Jpgraph 类库创建饼形图，分析 2010 年图书销量，旨在突出饼形图的创建方法。

（1）创建饼形图图像应用 PieGraph 类，它继承 Graph 类，创建 3D 饼形图的图像。





（2）创建 3D 饼形图应用 PiePlot3D 类，继承 PiePlot 类。

（3）通过图像中的 legend 对象调用 Pos()方法定位饼形图在图像中的存储位置。Pos()方法的语法如下：

```
Pos($aX, $aY, $aHAlign, $aVAlign)
```

参数说明：

☑ $aX：设置距离 X 轴的位置。

☑ $aY：设置距离 Y 轴的位置。

☑ $aHAlign：设置右侧（居中、居左或者居右）。

☑ $aVAlign：设置顶部（居下、居上或者居中）。

（4）通过饼形图对象调用 SetTheme()方法设置饼形图的颜色。SetTheme()方法的语法如下：

```
PiePlot ::
SetTheme($aTheme)
```

参数$aTheme 指定主题的名称，其支持的主题包括“earth”、“pastel”、“sand”、“water”。

多学两招：

在 Jpgraph 类库中，通过饼形图对象中 SetCenter()方法控制饼形图的大小比例，默认值是 0.5；通过 SetAngle()方法控制图像的倾斜角度，倾斜角度必须在 10°～80°之间。

实例 181　GD2 函数折线图分析网站月访问量走势

（实例位置：配套资源\SL\10\181）

实例说明

本实例运用 GD2 函数自行编写一个绘制折线图的方法，对 2010 年公司网站的月访问量进行分析。其中的方法完全由笔者自行编写，不借助于任何图像的操作类库，运行结果如图 10.10 所示。

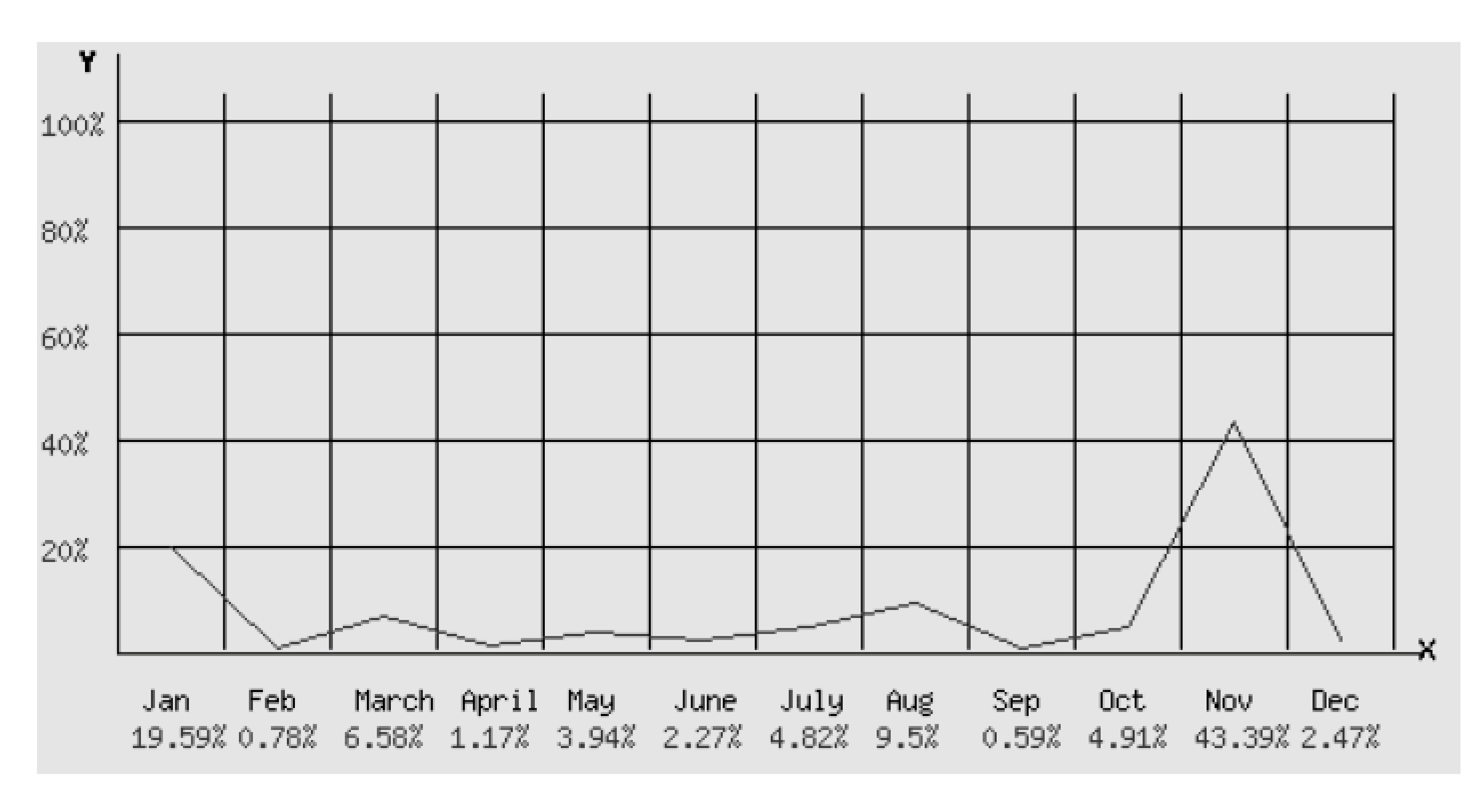


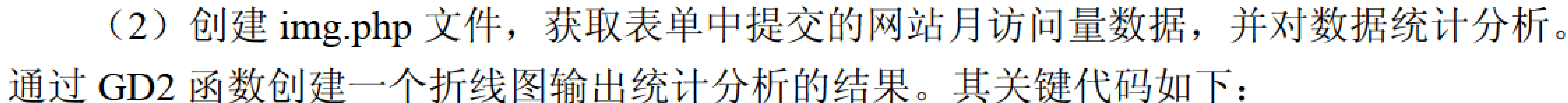
图 10.10　GD2 函数折线图分析网站月访问量走势



实现过程


具体步骤如下：


（1）创建 index.php 文件，定义表单，添加 12 个表单元素，提交网站的月访问量数据。


（2）创建 img.php 文件，获取表单中提交的网站月访问量数据，并对数据统计分析。通过 GD2 函数创建一个折线图输出统计分析的结果。其关键代码如下：

```
<?php
$data = array ($_POST ["T1"], $_POST ["T2"], $_POST ["T3"], $_POST ["T4"], $_POST ["T5"],
$_POST ["T6"], $_POST ["T7"], $_POST ["T8"], $_POST ["T9"], $_POST ["T10"], $_POST ["T11"], $_POST
["T12"] );
$month = array ("Jan", "Feb", "March", "April", "May", "June", "July", "Aug", "Sep", "Oct", "Nov",
"Dec" );
$max = 0;
for($i = 0; $i < 12; $i ++) {
    $max = $max + $data [$i];                              //所有网站访问量的累加和
}
$im = imagecreate ( 550, 300 );                            //创建画布
$green = imagecolorallocate ( $im, 214, 235, 214 );        //设置颜色值
$black = imagecolorallocate ( $im, 0, 0, 0 );
$red = imagecolorallocate ( $im, 255, 0, 0 );
$blue = imagecolorallocate ( $im, 0, 0, 255 );
imageline ( $im, 30, 230, 520, 230, $blue );               //设置 X 轴横坐标
imageline ( $im, 30, 5, 30, 230, $blue );                  //设置 Y 轴纵坐标
imagestring ( $im, 3, 520, 222, "X", $black );             //输出字符 X
imagestring ( $im, 3, 16, 1, "Y", $black );                //输出字符 Y
$l = 190;
$k1 = 30;
$k2 = 510;
for($j = 0; $j < 12; $j ++) {
    imageline ( $im, $k1, $l, $k2, $l, $black );           //设置 X 轴网格线横坐标
    $l = $l - 40;
}
$f = 70;
$z1 = 20;
$z2 = 228;
for($j = 0; $j < 12; $j ++) {
    imageline ( $im, $f, $z1, $f, $z2, $black );           //设置 Y 轴网格线纵坐标
    $f = $f + 40;
}
$l = 185;
for($j = 1; $j < 6; $j ++) {
    imagestring ( $im, 2, 2, $l, 20 * $j . "%", $red );
    $l = $l - 40;
}
$x = 20;
```





```
        $y = 230;
        for($i = 1; $i < 12; $i ++) {
            $y_lt = $y - (($data [$i - 1] / $max) * 200);          //设置网站访问量的纵坐标值
            $y_ht = $y - (($data [$i] / $max) * 200);              //获取每月网站访问量数的纵坐标值
        imageline ( $im, $x * ($i * 2 - 1) + 30, $y_lt, $x * (($i + 1) * 2 - 1) + 30, $y_ht, $red );   //绘制网站访
问量折线图
        }
        for($i = 1; $i < 13; $i ++) {
            $r1 = round ( (($data [$i - 1]) / $max) * 100, 2 );
            imagestring ( $im, 2, $x * ($i - 1) * 2 + 40, $y + 11, $month [$i - 1], $black ); //输出月份的值
          imagestring ( $im, 2, $x * ($i - 1) * 2 + 36, $y + 25, $r1 . "%", $red ); //输出网站访问量的百分比
        }
        imagepng ( $im );
        imagedestroy ( $im );                                       //释放图像资源
```



技术要点

本实例通过 GD2 函数绘制网站访问量走势分析图，其中所涉及的关键技术如下：

（1）应用 imageline()函数绘制线条。

imageline()函数用指定颜色在图像中两点之间绘制一条线段，语法如下：

```
bool imageline ( resource image, int x1, int y1, int x2, int y2, int color )
```

参数说明：

☑ image：图像标识。
☑ x1：起始点的横坐标。
☑ y1：起始点的纵坐标。
☑ x2：结束点的横坐标。
☑ y2：结束点的纵坐标。
☑ color：颜色标识。

（2）应用百分比来显示数据信息，并通过 round()函数将数字小数位进行四舍五入操作，round()函数语法如下：

```
double round(double val)
```

参数 var 为进行四舍五入操作的小数。

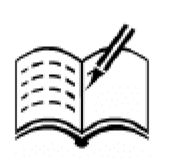
实例 182　GD2 函数柱状图分析编程词典满意度调查

（实例位置：配套资源\SL\10\182）

实例说明

在网站中，经常需要对相关的信息进行调查统计，然后根据访问者的投票结果制定相关计划。为了能够更直观地查看访问者的投票结果，本实例采用柱形图显示编程词典满意度的投票结果，运行结果如图 10.11 所示。





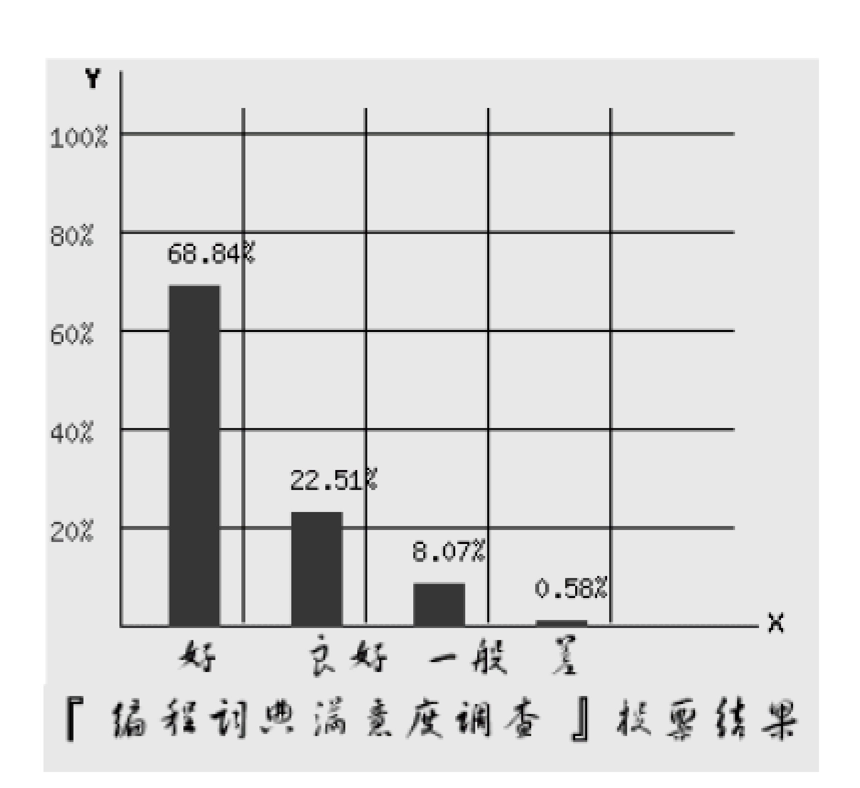




图 10.11　GD2 函数柱状图分析编程词典满意度调查

实现过程

具体步骤如下：

（1）创建 index.php 页面，编写 img 标签输出 img.png 图像，即输出柱形图统计结果。

（2）创建 img.php 文件，从数据库中读取数据，将从数据库中读取的数据以柱形图形式进行输出。其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
include ("conn/conn.php");
$query = mysql_query ( "select sum(ticket) as total from tb_comment" );
$info = mysql_fetch_array ( $query );
$max = $info [total];                                               //所有投票的累加和
$sql = mysql_query ( "select * from tb_comment" );
$result = mysql_fetch_array ( $sql );                               //检索数据信息
$im = imagecreate ( 350, 300 );                                     //创建画布
$green = imagecolorallocate ( $im, 233, 240, 151 );                 //设置颜色值
$black = imagecolorallocate ( $im, 0, 0, 0 );
$red = imagecolorallocate ( $im, 255, 0, 0 );
$blue = imagecolorallocate ( $im, 0, 0, 255 );
imageline ( $im, 30, 230, 290, 230, $blue );                        //设置 X 轴横坐标
imageline ( $im, 30, 5, 30, 230, $blue );                           //设置 Y 轴纵坐标
imagestring ( $im, 3, 295, 222, "X", $black );                      //输出字符 X
imagestring ( $im, 3, 16, 1, "Y", $black );                         //输出字符 Y
$l = 190;
$k1 = 30;
$k2 = 280;
for($j = 0; $j < 5; $j ++) {
  imageline ( $im, $k1, $l, $k2, $l, $black );                      //设置 X 轴网格线横坐标
  $l = $l - 40;
}
$f = 80;
$z1 = 20;
$z2 = 228;
for($j = 0; $j < 4; $j ++) {
  imageline ( $im, $f, $z1, $f, $z2, $black );                      //设置 Y 轴网格线纵坐标
  $f = $f + 50;
```





```
}
$l = 185;
for($j = 1; $j < 6; $j ++) {
  imagestring ( $im, 2, 2, $l, 20 * $j . "%", $red );
  $l = $l - 40;
}
$x = 45;
$y = 230;
$x_width = 20;
$y_ht = 0;
do {
        if ($result [comment] == "好") {
              $comment = "好";                                    //定义输出字体串
        }
        if ($result [comment] == "良好") {
              $comment = "良好";
        }
        if ($result [comment] == "一般") {
              $comment = "一般";
        }
        if ($result [comment] == "差") {
              $comment = "差";                                    //定义输出字体串
        }
        $fnt = "Font/FZHCJW.TTF";                                 //定义字体
        $y_ht = round ( ($result [ticket] / $max) * 200, 2 );     //设置投票结果所占百分比
        $y_ht1 = round ( ($result [ticket] / $max) * 100, 2 );    //设置投票结果所占百分比
        imagefilledrectangle ( $im, $x, $y, $x + $x_width, $y - $y_ht, $red ); //绘制并填充柱形图
  imagestring ( $im, 2, $x + $x_width - 20, $y - $y_ht - 20, $y_ht1 . "%", $black ); //设置投票结果所
占百分比
        imageTTFText ( $im, 15, 0, $x + 2, $y + 20, $black, $fnt, $comment ); //写 TTF 文字到图中
        $x += ($x_width + 30);                      //设置两柱形图之间的宽度为 30 像素
} while ( $result = mysql_fetch_array ( $sql ) );
$text = "『 编程词典满意度调查 』投票结果";
imageTTFText ( $im, 15, 0, 10, $y + 50, $black, $fnt, $text );     //写 TTF 文字到图中
imagedestroy ( $im );                                              //释放图像资源
?>
```

指点迷津：

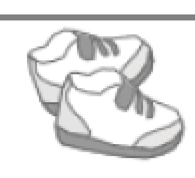
本实例运用的是 UTF-8 编码格式，所以在图像中输出中文字符串时可以直接应用 imageTTFText()函数；但是，如果页面使用的是 GB2312 编码格式，那么要输出中文字符串时，就必须将中文字符串转换成 UTF-8 编码格式，否则将输出乱码。

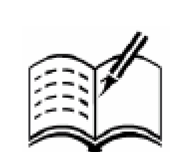
技术要点

通过 GD2 函数创建柱形图，关键是 imagefilledrectangle()函数，通过它完成柱形图的绘制及填充。

imagefilledrectangle()函数，在图像中绘制一个用指定颜色填充的矩形，其语法如下：

```
bool imagefilledrectangle ( resource image, int x1, int y1, int x2, int y2, int color )
```





参数说明：

☑ image：图像资源。
☑ x1：柱状图左上角横坐标。
☑ y1：柱状图左上角纵坐标。
☑ x2：柱状图右下角横坐标。
☑ y2：柱状图右下角纵坐标。
☑ color：填充颜色。

在本实例中，将从数据库中读取的数据以柱形图的形式进行展示，实现柱形图与数据库中数据的完美结合。

实例183　GD2函数饼形图分析图书市场的份额

（实例位置：配套资源\SL\10\183）

实例说明

在调查某类商品的市场占有率的时候，最好的显示方式就是使用饼形图。通过饼形图可以直观地看到某类产品的不同品牌在市场中的占有比例。运行本实例，通过饼形图将不同语言的软件图书在市场中的占有率显示出来，运行结果如图10.12所示。

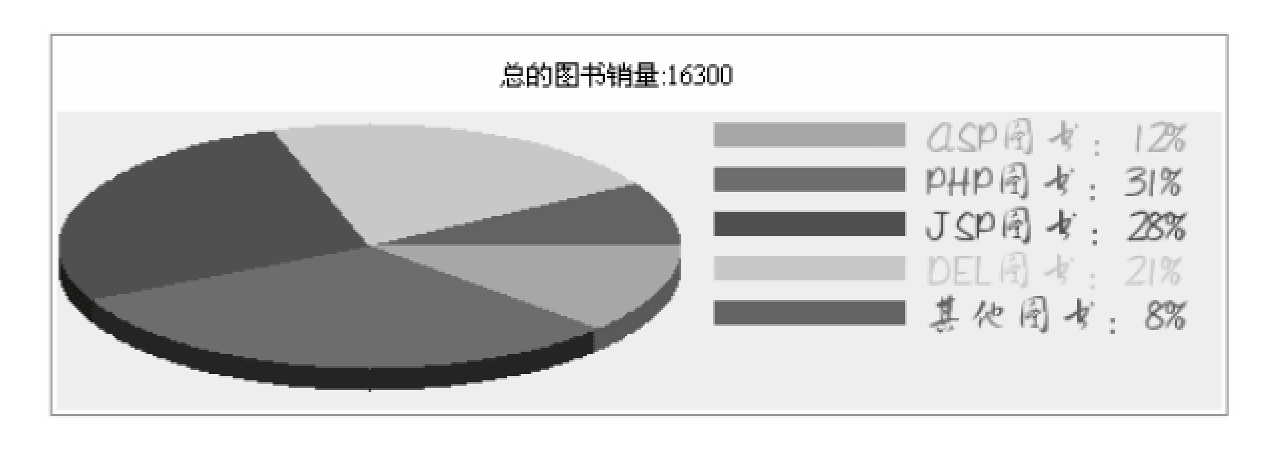


图10.12　GD2函数饼形图分析图书市场的份额

实现过程

具体步骤如下：

（1）创建index.php文件，连接数据库，执行查询语句，获取数据库中总的图书销量值，并通过img标签调用图像处理文件caky_img.php。

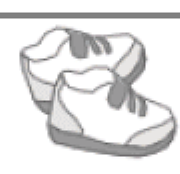
（2）创建caky_img.php文件，通过饼形图和柱形图联合分析各语言图书市场的份额。首先，获取表单中传递的总的图书销量值，并计算各部门图书销量值占总图书销量的比例。然后，根据比例值创建饼形图。最后，通过柱形图输出不同的颜色区块代表相应语言的图书。其关键代码如下：

```
<?php
include ("conn/conn.php");                                          //连接数据库
$query = mysql_query ( "select product_count from tb_data " );      //执行查询语句
$data = array ();                                                   //定义空数组
while ( $myrow = mysql_fetch_array ( $query ) ) {
  $data [] = $myrow [product_count];                                //将查询结果写入到数组中
}
```






```
        $widths = 580;                                          //定义图像宽度
        $heights = 150;                                         //定义图像高度
        $width = 280;                                           //定义饼形图宽度
        $height = 130;                                          //定义饼形图高度
        $angle = array ();                                      //角度
        $total = 0;
        for($i = 0; $i < count ( $data ); $i ++) {
            if (! is_numeric ( $data [$i] ))                    //判断数组中的值是否为空
                die ( "1" );
            $total += $data [$i];
        }
        for($i = 0; $i < count ( $data ); $i ++) {
            array_push ( $angle, round ( 360 * $data [$i] / $total ) );   //定义饼形图的角度值
        }
        $image = imagecreate ( $widths, $heights );             //创建饼形图的画布
        $white = imagecolorallocate ( $image, 0xEE, 0xEE, 0xEE );   //定义背景颜色
        $color = array (array (imagecolorallocate ( $image, 0x97, 0xbd, 0x00) );   //定义图像的颜色值
        $rounds_x = $width / 2;                                         //原线
        for($h = $height / 2 + 5; $h > $height / 2 - 5; $h --) {
            $start = 0;
            $stop = 0;
            for($i = 0; $i < count ( $data ); $i ++) {
                $start = $start + 0;
                $stop = $start + $angle [$i];
                $color_i = fmod ( $i, 10 );
                imagefilledarc ( $image, $rounds_x, $h, $width, $height - 20, $start, $stop, $color [1]
[$color_i], IMG_ARC_PIE );                                      //创建饼形图
                $start += $angle [$i];
                $stop += $angle [$i];
            }
        }
        39      for($i = 0; $i < count ( $data ); $i ++) {
            $start = $start + 0;
            $stop = $start + $angle [$i];
            $color_i = fmod ( $i, 10 );
            imagefilledarc ( $image, $rounds_x, $h, $width, $height - 20, $start, $stop, $color [0]
[$color_i], IMG_ARC_PIE );
            $start += $angle [$i];
            $stop += $angle [$i];
        }
        $fnt = "Font/FZHCJW.TTF";                               //定义字体
        $m = - 1;
        $result = mysql_query ( "select * from tb_data " );     //读取数据库中存储的数据
        while ( $myrow = mysql_fetch_array ( $result ) ) {      //while 循环输出数据库中数据
            $m ++;
          imagefilledrectangle ( $image, 295, 15 + 20 * $m, 380, 5 + 20 * $m, $color [0] [$m] ); //绘制并填充
柱形图
            $comment = $myrow [product_name] . "：" . number_format ( $myrow [product_count] /
$_GET [counts], 2 ) * 100 . "%";                                //输出柱形图
          imageTTFText ( $image, 15, 0, 390, 18 + 20 * $m, $color [0] [$m], $fnt, $comment ); //写 TTF 文字
到图中
```



```
    }
    imagepng ( $image );
    imagedestroy ( $image );
    ?>
```



Note

（3）创建 conn 文件夹，定义 conn.php 文件，连接 MySQL 数据库服务器，连接 db_database10 数据库，并设置数据库编码格式为 UTF-8。

技术要点

（1）绘制椭圆使用 GD2 函数库中的 imagefilledarc()函数，该函数在指定图像标识所代表的图像中以图像左上角为坐标点绘制一椭圆弧。如果成功，则返回 true；否则，返回 false。该函数的语法如下：

```
bool imagefilledarc ( resource image, int cx, int cy, int w, int h, int s, int e, int color, int style )
```

参数说明：

- ☑ image：图像标识。
- ☑ cx：图像左上角横坐标。
- ☑ cy：图像左上角纵坐标。
- ☑ w：椭圆的宽度。
- ☑ h：椭圆的高度。
- ☑ s：以角度指定椭圆的起始点。
- ☑ e：以角度指定椭圆的结束点。
- ☑ color：颜色标识。
- ☑ style：可以是 IMG_ARC_PIE、IMG_ARC_CHORD、IMG_ARC_NOFILL、IMG_ARC_EDGED 按位或（OR）后的值，IMG_ARC_PIE 和 IMG_ARC_CHORD 是互斥的。IMG_ARC_CHORD 只是用直线连接了起始和结束点，IMG_ARC_PIE 则产生圆形边界（如果两个都用，IMG_ARC_CHORD 生效）。IMG_ARC_NOFILL 指明弧或弦只有轮廓，不填充。IMG_ARC_EDGED 指明用直线将起始和结束点与中心点相连，和 IMG_ARC_NOFILL 一起使用是绘制饼状图轮廓的好方法（而不用填充）。

（2）同时应用绘制柱形图的方法，通过柱形图展示不同颜色代表的数据。有关柱形图的应用请参考实例 182。

（3）充分展示数据库中数据与图形结合的方法。分别运用 for 语句和 while 语句对数据库中的数据进行循环输出。

实例 184　柱状图展示编程词典 6、7 月份销售量

（实例位置：配套资源\SL\10\184）

实例说明

本实例运用 Jpgraph 生成对比柱状图，对编程词典 6、7 月份的销售量进行比较，运行



结果如图 10.13 所示。

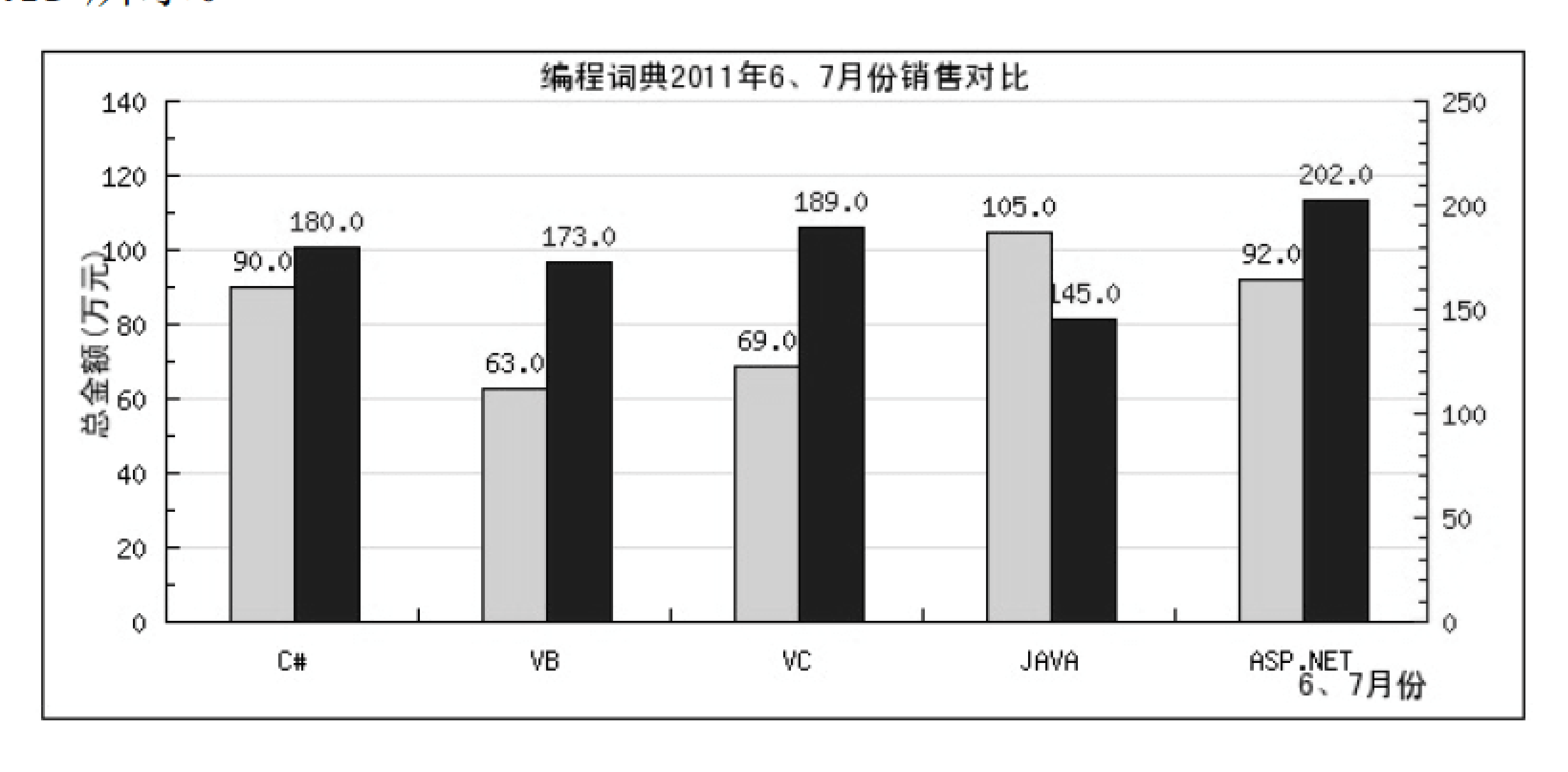


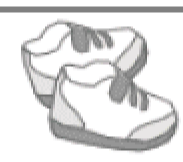
图 10.13　柱状图展示编程词典 6、7 月份销量

实现过程

（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

（2）应用 Jpgraph 类库中的方法完成对比柱形图的创建，其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );             //设置文件编码格式
include ("../src/jpgraph.php");                                  //导入文件
include ("../src/jpgraph_bar.php");                              //导入文件
$data_july = array(90, 63, 69, 105, 92);                         //设置统计数据
$data_june = array(180, 173, 189, 145, 202);                     //设置统计数据
$datazero=array(0,0,0,0,0);
$datas = array("C#", "VB", "VC", "JAVA", "ASP.NET");             //设置统计数据
$graph = new Graph(600,270);                                     //创建图像
$graph->SetScale("textlin");                                     //设置坐标刻度类型
$graph->SetY2Scale("lin");
$graph->img->SetMargin(50, 40, 20, 40);                          //设置统计图边距
$graph->yaxis->scale->SetGrace(20);
$graph->y2axis->scale->SetGrace(20);
$graph->SetMarginColor('white');                                 //创建背景颜色
$graph->y2axis->SetColor('darkred');
$bplotzero = new BarPlot($datazero);                             //设置图像的填充值
$ybplot1 = new BarPlot($data_july);                              //创建 6 月份数据
$ybplot1->value->Show();
$ybplot = new GroupBarPlot(array($ybplot1,$bplotzero));
$ybplot2 = new BarPlot($data_june);                              //创建 7 月份数据
$ybplot2->value->Show();                                         //输出数据
$ybplot2->value->SetColor('darkred');
$ybplot2->SetFillColor('darkred');
$y2bplot = new GroupBarPlot(array($bplotzero,$ybplot2));
//将数据添加到图像中
$graph->Add($ybplot);
$graph->AddY2($y2bplot);
```




```
$graph->title->Set(iconv("utf-8","gb2312",'编程词典 2011 年 6、7 月份销售对比')); //设置统计图标题
$graph->xaxis->title->Set(iconv("utf-8","gb2312",'6、7 月份'));          //设置 X 轴名称
$graph->xaxis->SetTickLabels($datas);
$graph->yaxis->title->Set(iconv("utf-8","gb2312",'总金额(万元)'));       //设置 Y 轴名称
$graph->title->SetFont(FF_SIMSUN, FS_BOLD);                            //设置标题字体
$graph->xaxis->title->SetFont(FF_SIMSUN,FS_BOLD);                      //设置 X 轴字体
$graph->yaxis->title->SetFont(FF_SIMSUN,FS_BOLD);                      //设置 Y 轴字体
$graph->Stroke();                                                      //生成图像
?>
```



Note

技术要点

本实例通过柱状图完成对编程词典 6、7 月份销量的对比。它在单一柱状图的基础上增加一组柱状图，形成两组数据的对比效果，其实现方法与单一柱状图是相同的，可以说是创建两个单一的柱状图，然后将其在同一图像中输出。

（1）通过 BarPlot 类创建柱状图，运用其返回对象调用 value()方法，调用 value()方法中的 Show()方法输出数据，通过 SetColor()方法设置输出数据颜色。

SetColor()方法的语法如下：

```
SetColor($aColor, $aLabelColor)
```

☑ $aColor：设置坐标轴的颜色。

☑ $aLabelColor：默认值为 false，设置坐标轴上标签的颜色。

（2）通过 Add()方法将柱状图数据添加到图像中，其参数是柱状图创建返回的对象。语法如下：

```
Graph ::
Add($aPlot)
```

参数$aPlot 指定图像创建时返回的对象。

（3）在本实例中，还应用到了 GroupBarPlot 类创建柱状图数组，通过该类可以实现对柱状图的分段处理，如果重新设置数组$datazero 的值，就会发现柱状图有一些变化。

```
Class GroupBarPlot Extends BarPlot
GroupBarPlot ::
GroupBarPlot($plots)
```

GroupBarPlot 创建一个新的图像对象，$plots 是一个数组。

实例 185　柱状图展示编程词典上半年销量

（实例位置：配套资源\SL\10\185）

实例说明

本实例运用 Jpgraph 生成柱状图，对公司编程词典上半年的销量进行统计，运行结果如图 10.14 所示。



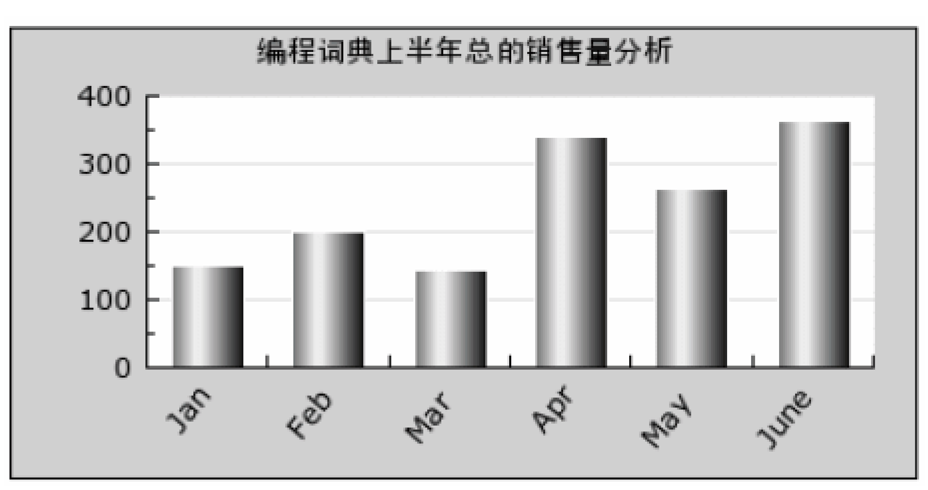
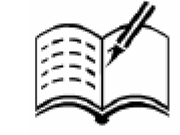


图 10.14　柱状图展示编程词典上半年销量



实现过程

具体步骤如下：

（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

（2）应用 Jpgraph 类库中的方法完成特殊柱状图的创建，其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                    //设置文件编码格式
include ("../src/jpgraph.php");
include ("../src/jpgraph_bar.php");
$datay=array(150,201,145,340,265,365);
$datax=array("Jan","Feb","Mar","Apr","May","June");
$graph = new Graph(400,200,"auto");                                     //创建图像
$graph->img->SetMargin(60,20,30,50);                                    //设置图像边框间距
$graph->SetScale("textlin");                                            //定义坐标刻度类型
$graph->SetMarginColor("lightblue");                                    //定义图像颜色
$graph->title->Set(iconv("utf-8","gb2312",'编程词典上半年总的销售量分析'));  //定义标题
$graph->title->SetFont(FF_SIMSUN, FS_BOLD);                             //设置标题字体
$graph->xaxis->SetFont(FF_VERDANA,FS_NORMAL,10);                        //设置 X 轴的字体
$graph->yaxis->SetFont(FF_VERDANA,FS_NORMAL,10);                        //设置 Y 轴的字体
$graph->xaxis->SetTickLabels($datax);                                   //设置 X 轴输出的数据
$graph->xaxis->SetLabelAngle(50);                                       //设置输出文字大小
$bplot = new BarPlot($datay);                                           //实例化图像创建类
$bplot->SetWidth(0.6);                                                  //设置柱形图的输出大小
$bplot->SetFillGradient("navy","#FFFF00",GRAD_LEFT_REFLECTION);//设置图像的类型和填充
颜色
$bplot->SetColor("white");                                              //设置图像边框颜色
$graph->Add($bplot);                                                    //添加数据
$graph->Stroke();                                                       //生成图像
?>
```

技术要点

本实例对编程词典上半年的销量进行分析，这里创建的柱状图与上面创建的柱状图唯一的区别是形状和颜色不同。

在本实例中，设置柱状图形状是通过 BarPlot 类中的 SetFillGradient()方法完成颜色、形状的设置，其语法如下：

```
SetFillGradient($aFromColor, $aToColor, $aStyle)
```

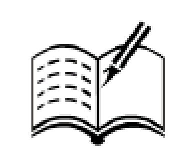


参数说明：

☑ $aFromColor：默认颜色。

☑ $aToColor：转换的颜色。

☑ $aStyle：设置的样式。



本实例中的具体应用如下：

```
SetFillGradient ("navy","#FFFF00",GRAD_LEFT_REFLECTION);    //设置图像的类型和填充颜色
```

实例 186　折线图分析 2010 年牛肉市场价格走势

（实例位置：配套资源\SL\10\186）

实例说明

本实例运用 Jpgraph 生成折线图，对 2010 年牛肉市场价格走势进行分析，运行结果如图 10.15 所示。

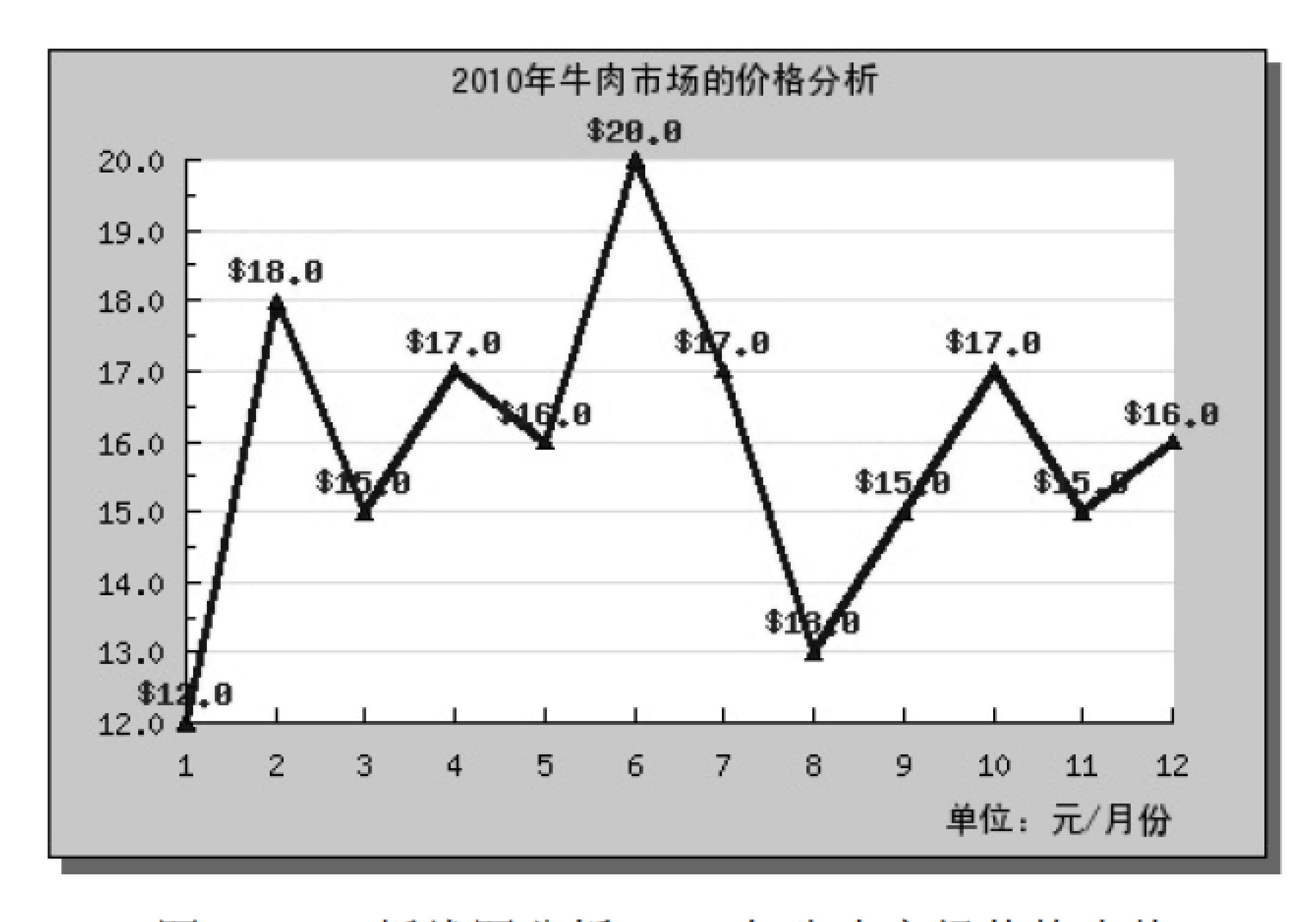


图 10.15　折线图分析 2010 年牛肉市场价格走势

实现过程

具体步骤如下：

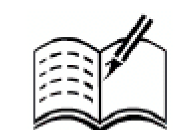
（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

（2）应用 Jpgraph 类库中的方法创建一个折线图，对 2010 年牛肉市场价格走势进行分析。其代码如下：

```
$ydata = array(12,18,15,17,16,20,17,13,15,17,15,16);          //定义数据
$graph = new Graph(450,300,"auto");                           //创建图像
$graph->SetScale("textlin");                                  //定义刻度值类型
$graph->img->SetMargin(50,40,40,55);                          //设置边框的间距
$lineplot=new LinePlot($ydata);                               //创建折线图
$lineplot->mark->SetType(MARK_UTRIANGLE);                     //定义折线图的标记点
$lineplot->value->show();                                     //输出折线图对应的值
```



```
$lineplot->value->SetColor('darkred');                                  //设置值的颜色
$lineplot->value->SetFont(FF_FONT1,FS_BOLD);                            //设置值的字体
$lineplot->value->SetFormat('$%0.1f');                                  //对值进行格式化
$graph->Add($lineplot);                                                 //添加数据
$graph->title->Set(iconv("utf-8","gb2312","2010 年牛肉市场的价格分析"));//定义标题
$graph->xaxis->title->Set(iconv("utf-8","gb2312","单位：元/月份"));     //定义 X 轴输出的值
$graph->xaxis->title->SetMargin(10);                                    //定义 X 轴输出的值
$graph->title->SetFont(FF_SIMSUN, FS_BOLD);                             //设置标题字体
$graph->yaxis->title->SetFont(FF_SIMSUN, FS_BOLD);                      //设置 X 轴内容的字体
$graph->xaxis->title->SetFont(FF_SIMSUN, FS_BOLD);                      //设置 Y 轴内容的字体
$lineplot->SetColor("blue");                                            //设置图像颜色
$lineplot->SetWeight(2);                                                //设置折线的宽度
$graph->Stroke();                                                       //生成图像
?>
```



技术要点

本实例通过 Jpgraph 类库创建折线图，分析 2010 年牛肉市场的价格走势，旨在突出如何在折线图中输出格式化的数据，以及在折线图的数据点上添加标记。

（1）输出数据应用的是 value()对象中的 Show()方法，数据的格式化应用的是 SetFormat()方法。

```
SetFormat($aFormat, $aNegFormat)
```

参数说明：

☑ $aFormat：设置格式化的样式。

☑ $aNegFormat：设置 negative 的值。

（2）折线图中标记的添加应用的是图像中的 mark 对象的 SetType()方法，其参数值设置标记的类型。其语法如下：

```
SetType($aType, $aFileName, $aScale)
```

参数说明：

☑ $aType：设置标记的类型。

☑ $aFileName：插入图像的名称或者国家的名称。

☑ $aScale：设置插入标记的大小。

下面列举了一些 SetType()方法中第一个参数可选的类型：MARK_SQUARE、MARK_UTRIANGLE、MARK_DTRIANGLE、MARK_DIAMOND、MARK_CIRCLE，这些标记仅供参考，如果想了解更多内容请参考 Jpgraph 类库的参考手册。

实例 187　多饼形图区块分析 2010 年图书销量

（实例位置：配套资源\SL\10\187）

实例说明

本实例运用 Jpgraph 生成饼形图，对公司 2010 年图书销量进行分析，其运行结果如

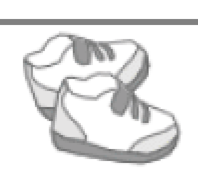
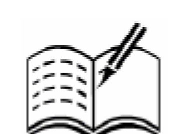


图 10.16 所示。



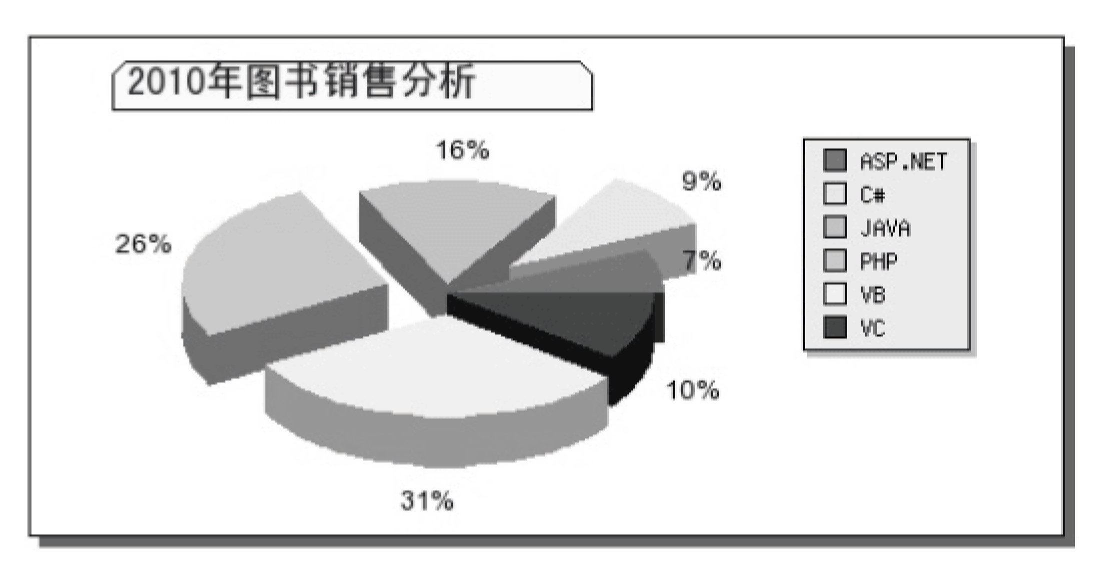


图 10.16　多饼形图区块分析 2010 年图书销量

实现过程

具体步骤如下：

（1）创建 index.php 文件，设置网页的编码格式，并通过 include()语句导入所需的存储在章文件夹 10 下的 Jpgraph 文件。

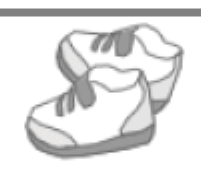
（2）应用 Jpgraph 类库中的方法创建一个饼形图，对 2010 年图书销量进行分析。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                    //设置文件编码格式
include ("../src/jpgraph.php");
include ("../src/jpgraph_pie.php");
include ("../src/jpgraph_pie3d.php");
$data = array(20,27,45,75,90,30);
$graph = new PieGraph(500,245,"auto");                                  //创建图像
$graph->SetShadow();                                                    //创建图像阴影
$graph->tabtitle->Set(iconv("utf-8","gb2312",'2010 年图书销售分析') );   //输出标题
$graph->tabtitle->SetFont(FF_SIMSUN, FS_BOLD,14);                       //设置标题字体
$graph->title->SetColor("darkblue");                                    //定义标题颜色
$graph->legend->Pos(0.1,0.2);                                           //控制注释文字的位置
$p1 = new PiePlot3d($data);                                             //创建图像
$p1->SetTheme("sand");                                                  //控制图像颜色
$p1->SetCenter(0.4);                                                    //设置图像位置
$p1->SetSize(0.4);                                                      //设置图像大小
$p1->SetHeight(20);                                                     //设置饼形图高度
$p1->SetAngle(45);                                                      //设置图像倾斜角度
$p1->Explode(array(5,40,10,30,20));                                     //控制饼形图的分割
$p1->value->SetFont(FF_ARIAL,FS_NORMAL,10);                             //设置字体
$p1->SetLegends(array("ASP.NET","C#","JAVA","PHP","VB","VC"));
$graph->Add($p1);                                                       //添加数据
$graph->Stroke();                                                       //生成图像
?>
```

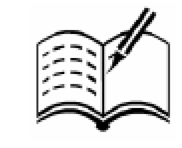
技术要点

本实例通过 Jpgraph 类库创建饼形图，应用饼形图对象中的 Explode()方法完成对 2010 年各部门图书销量的分块展示。Explode()方法语法如下：

```
Explode($aExplodeArr)
```



将饼形图分割成多个模块，参数$aExplodeArr 是一个数组，根据数组元素个数定义分割成几块，根据数组元素值定义分割后模块与原图像的距离。



实例 188　缩略图艺术库

（实例位置：配套资源\SL\10\188）

实例说明

上传图片是一个比较常见的功能，但是在输出上传图片时可能会遇到一些问题。例如，由于上传图片的大小没有固定，从而导致图片在输出时变形，这是一个很让人伤脑筋的问题。如果在上传图片时直接将其生成一个固定大小的缩略图，并同时保存上传的原始图片，那么在输出时就不会有任何问题。

本实例就实现了这样一个功能，在将图片上传到服务器的同时，生成图片的缩略图，直接浏览图片的缩略图。如果要下载，则下载的仍是原始图片。本实例的运行效果如图 10.17 所示。



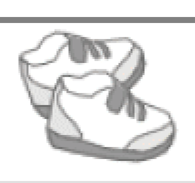
图 10.17　缩略图艺术库

实现过程

具体步骤如下：

（1）在上传文件的处理页面 index_ok.php 中，包含了一个 thumbnail.php 文件；重新定义上传文件的类型，并重新指定上传文件的存储位置，定义原始文件存储文件夹和缩略图存储文件夹；在通过 move_uploaded_file()函数完成文件的上传操作后，调用 thumbnail.php 文件中包含的方法 makethumb()将上传的图片生成缩略图，并添加文字水印。这是在 index_ok.php 文件中所作的处理，其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                    //设置文件编码格式
include "conn/conn.php";
include "thumbnail.php";
```






```
        if ($_POST [files] != "") {
          $data = date ( "Y-m-d H:m:s" );
          function check($var) {                                                //验证数组的返回值是否为空
                  return ($var != "");
          }
          $files = array_filter ( $_POST ["files"], "check" );                  //去除数组中空值
          $array = array_filter ( $_FILES ["picture"] ["name"], "check" );      //去除数组中空值
          foreach ( $array as $key => $value ) {                                //循环读取数组中数据
                  $types = strtolower ( strstr ( $value, '.' ) );               //截取上传文件的后缀
                  if ($types == ".jpg" || $types == ".gif" || $types == ".png") { //判断上传文件的后缀是否符合
要求
                          $thumbnail = 'thumbnail/';
                          $original = 'original/';
                          $file_name = time () . $key . strtolower ( strstr ( $value, "." ) ); //定义上传文件名称
                          $path = $original . $file_name;                       //定义原始文件的存储位置
                          $path_thumbnail = $thumbnail . $file_name;            //定义缩略图的存储位置
                          move_uploaded_file ( $_FILES ["picture"] ["tmp_name"] [$key], $path );//执行上传操作
                          makethumb ( $path, $path_thumbnail, "200", "200", "100", "www.mrbccd.com" );
//生成缩略图
                          $query = "insert into tb_up_file (file_test,path_thumbnail,data,file_name) values
('$path','$path_thumbnail','$data','$files[$key]')";
                          $result = mysql_query ( $query );
                          echo "<script>alert('图片上传成功'); window.location.href='index.html';</script>";
                  } else {
                          echo "上传文件" . $value . "类型不正确！ ";
                  }
          }
        }
        ?>
```

（2）创建 thumbnail.php 文件，定义 makethumb()方法，完成缩略图的生成和水印的添加操作。自定义方法 makethumb()的语法如下：

```
makethumb($srcFile, $dstFile, $dstW, $dstH, $rate = 100, $markwords = null, $markimage = null){}
```

参数说明：

☑ $srcFile：图片文件名。

☑ $dstFile：另存文件名。

☑ $markwords：水印文字。

☑ $markimage：水印图片。

☑ $dstW：图片保存宽度。

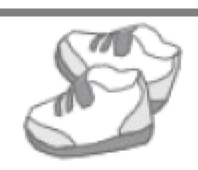
☑ $dstH：图片保存高度。

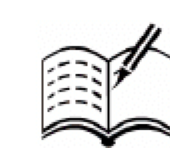
☑ $rate：图片保存品质，默认值 100。

有关 thumbnail.php 文件的完成代码请参考本书配套资源。

技术要点

本实例中重点讲解的是如何生成图片的缩略图，并为缩略图添加水印。其原理如下：





（1）通过 GetImageSize()函数获取上传图片的信息，包括大小、类型等信息，并根据其类型，新建一个对应类型的图像。

getimagesize()函数，获取指定图像的大小。语法如下：

```
array getimagesize ( string filename [, array &imageinfo] )
```

getimagesize()函数将测定任何 GIF、JPG、PNG、SWF、SWC、PSD、TIFF、BMP、IFF、JP2、JPX、JB2、JPC、XBM 或者 WBMP 图像文件的大小，并返回图像的尺寸以及文件类型和一个可以用于普通 HTML 文件中<IMG>标记中的 height/width 文本字符串。

返回值是一个具有四个单元的数组：

☑ 索引 0 包含图像宽度的像素值。
☑ 索引 1 包含图像高度的像素值。
☑ 索引 2 是图像类型的标记：1 = GIF，2 = JPG，3 = PNG，4 = SWF，5 = PSD，6 = BMP，7 = TIFF(intel byte order)，8 = TIFF(motorola byte order)，9 = JPC，10 = JP2，11 = JPX，12 = JB2，13 = SWC，14 = IFF，15 = WBMP，16 = XBM。
☑ 索引 3 是文本字符串，内容为“height="yyy" width="xxx"”，可直接用于 IMG 标记。

如果参数 filename 指定的图像或者其不是有效的图像，将返回 FALSE 并产生一条 E_WARNING 级的错误。

（2）对上传图片的大小与指定缩略图的大小进行比较，看其是否需要创建缩略图。

（3）根据判断结果应用 imagecopyresampled()函数重新生成新的图像。

imagecopyresampled()函数，从原图像中采样，复制部分图像，重新生成图像。语法如下：

```
bool imagecopyresampled ( resource dst_image, resource src_image, int dst_x, int dst_y, int src_x, int src_y, int dst_w, int dst_h, int src_w, int src_h )
```

imagecopyresampled()将一幅图像中的一块正方形区域复制到另一个图像中，平滑地插入像素值，在减小图像的大小的同时，仍然保持了极大的清晰度。

参数说明：

☑ dst_image：是目标图像标识符。
☑ src_image：是源图像标识符。

如果源图像和目标图像的宽度和高度不同，则会进行相应的图像收缩和拉伸。坐标指的是左上角。本函数可用来在同一幅图内部复制（如果 dst_image 和 src_image 相同的话）区域，但如果区域交迭的话则结果不可预知。

如果成功，则返回 true；失败，则返回 false。

脚下留神：
本函数需要 GD 库 2.0.1 或者更高版本的支持。

（4）为重新生成的图像添加水印，可以选择文字水印或者图像水印。

添加文字水印应用 ImageTTFText()函数，如果页面使用的不是 UTF-8 编码，那么还要对水印文字的编码格式进行转换，转换成 UTF-8 编码，因为 ImageTTFText()函数只支持 UTF-8 编码。



添加图像水印应用 imagecopy()函数，完成对图像的复制操作。

实例 189　通过图像显示投票统计结果

（实例位置：配套资源\SL\10\189）



Note

实例说明

在线调查中的投票系统相信大家都了解，本实例编写了一个简易的投票系统，并将投票结果以图像的形式进行输出。运行效果如图 10.18 所示。

我最喜爱的运动员	选项	投票结果
○	欧阳一	2/2
⊙	南宫一	3/2
	确认提交	

图 10.18　通过图像显示投票统计结果

实现过程

本实例的文件架构如图 10.19 所示。

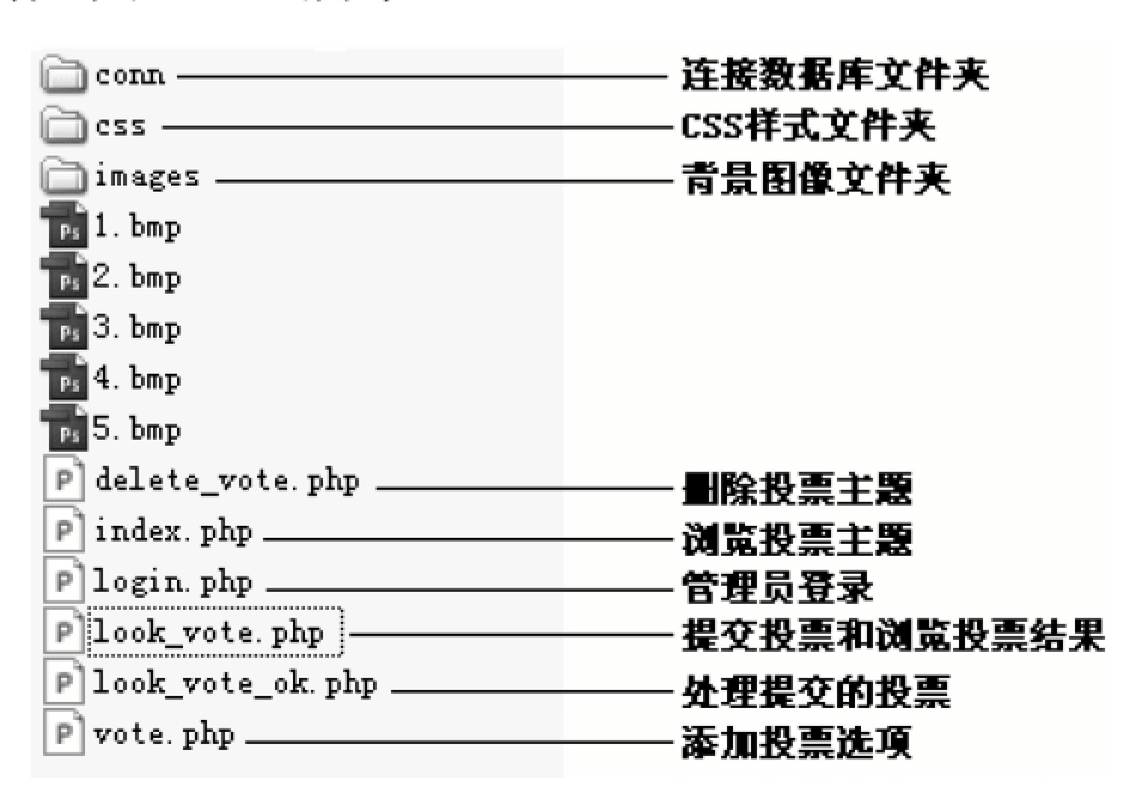
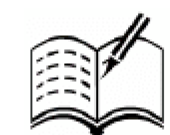


图 10.19　投票系统的文件架构

这里重点介绍 look_vote.php 文件，在该文件中完成投票的提交操作和投票结果的输出。首先，根据超链接传递的 ID，查询出指定的投票主题，并计算出投票总数。然后，创建表单，提交投票选项，将数据提交到 look_vote_ok.php 页进行处理。最后，执行查询语句，从数据库中查询出投票结果，并完成投票结果的输出。其关键代码如下：

```
<?php
include_once ("conn/conn.php");
if (! $_GET [id]) {
  echo "没有指定 ID！ ";
  exit ();
} else {
  ?>
//执行查询，输出投票主题
<?php
```



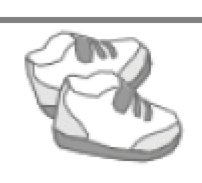


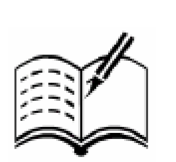
```
        $sql = "select * from tb_vote where tb_vote_id='$_GET[id]'"; //显示指定投票项
        $result = mysql_query ( $sql, $conn );
        $row = mysql_fetch_array ( $result );
        $s = $row [tb_vote_num];
        echo "$row[tb_vote_name]\n";
        ?>
    //创建表单，提交投票选项
    <form action=look_vote_ok.php method=post><input type=hidden name=id value="<?php echo $row
[tb_vote_id]; ?>"><input type=hidden name=v_type value="<?php echo $row [tb_vote_type]; ?>">
    </form>
    //输出投票结果
      <?php
      $sql2 = "select * from tb_vote_record where tb_vote_id='$_GET[id]'";
      $result2 = mysql_query ( $sql2, $conn );
      while ( $rows = mysql_fetch_array ( $result2 ) ) {
            ?>
    <tr>
                            <td height="25" align="center">
    <?php

            if ($row [tb_vote_type] == 0) {
                echo "<input type=radio name=r value=" . $rows [tb_record_id] . ">\n";
            } else {
                echo "<input type=checkbox name=r[] value=" . $rows [tb_record_id] . ">\n";
            }
            ?></td>
                            <td> <?php     echo $rows [tb_record_name]; ?></td>
                            <td> 
    <?php
            if ($rows [tb_record_num] == 0)
                $width = 0;
            else
                $width = $rows [tb_record_num] / $s;
            if ($width != 0) {
                echo "<img src=" . $rows [tb_record_color] . ".bmp width=" . ($width * 20) . " height=10>\n";
            }
            echo $rows [tb_record_num] . "/" . $row [tb_vote_num];
            ?></td>
                        </tr>
    <?php
      }
      ?>
    <?php
    }
    ?>
```

技术要点

投票系统包括：动态生成投票主题、动态添加投票选项内容、投票主题管理和投票功能实现。这里重点介绍在投票功能中如何以图像的形式输出投票结果。





Note

在 look_vote.php 文件中，根据总的投票数，计算出每一项所占的比例，然后根据这个比例值计算输出的图像，其关键代码如下：

```
<?php
$sql="select * from tb_vote where tb_vote_id='$_GET[id]'"; //显示指定投票项
$result=mysql_query($sql,$conn);                         //执行查询指定的投票选项
$row=mysql_fetch_array($result);                         //获取查询结果
$s=$row[tb_vote_num];                                    //计算总的投票数量
echo "$row[tb_vote_name]\n";                             //输出投票选项的名称
?>
<?php
  if($rows[tb_record_num]==0)                            //判断如果投票数为 0
  $width=0;                                              //定义变量值为 0
  else
  $width=$rows[tb_record_num]/$s;          //否则，根据投票数和总的投票数计算输出所占比例
  if($width!=0){                                         //以图像的形式输出投票所占比例
        echo "<img src=".$rows[tb_record_color].".bmp width=".($width*20)." height=10>\n";
  }
  echo $rows[tb_record_num]."/".$row[tb_vote_num];       //以数字输出投票数和总的投票量
?>
```

实例 190　通过图像显示密码安全强度

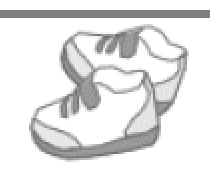
（实例位置：配套资源\SL\10\190）

实例说明

在本实例中，将介绍图像的另一项妙用，通过它来显示注册填写密码的安全强度。运行本实例，当在密码文本框中输入一个密码时，光标离开后，将自动判断这个密码的安全强度，并以图像的形式返回一个判断结果，运行结果如图 10.20 所示。



图 10.20　通过图像显示密码安全强度



Note

实现过程

本实例完成一个用户注册的功能，并且在表单中通过 JavaScript 脚本对提交的元素进行判断，在最终符合要求后，将用户注册信息添加到指定的数据表中。

（1）创建 index.php 页面，完成用户注册页面和表单的设计，通过 JavaScript 对表单中提交的值进行判断，并直接将判断的结果反馈到 div 标签中，最终将表单中的数据提交到 index_ok.php 文件中，完成用户注册的操作。

```
<form name="form_reg" method="post" action="index_ok.php" onSubmit="return chkreginfo(form_reg,'all')">
    <tr>
        <td height="30"><div align="right">密码：</div></td>
        <td height="30" colspan="2" align="left">  <input type="password" name="pwd" id="pwd" size="30" class="inputcss" onBlur="chkreginfo(form_reg,1)"> <font color="#FF0000">*</font>  
            <div id="chknew_pwd" style="color: #FF0000"></div>
        </td>
     </tr>
    <tr>
        <td height="30"> </td>
        <td width="150" height="30"><input type="image" src="images/form (2).jpg">   </td>
        <td width="343"><img src="images/form.jpg" width="72" height="26" onClick="form_reg.reset()" style="cursor: hand" /></td>
     </tr>
    </form>
```

（2）创建 JavaScript 脚本，编写 check.js 脚本文件，完成对表单中提交的值进行判断。

（3）创建 index_ok.php 文件，获取表单中提交的数据，将数据添加到指定的数据表中。

（4）创建 conn.php 文件，完成 MySQL 服务器和 MySQL 数据库的连接操作，并设置编码格式为 UTF-8。

技术要点

对密码的安全强度进行判断是在 JavaScript 脚本中应用正则表达式来完成的。其原理如下：

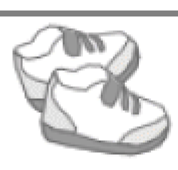
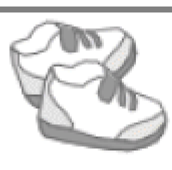
（1）对密码中的值进行判断，如果只包含数字、英文字符串或者特殊字符串中的任意一种，那么说明密码安全强度差。

（2）对密码中的值进行判断，如果包含数字、英文字符串或者特殊字符串中的任意两种，那么说明密码安全强度中。

（3）对密码中的值进行判断，如果同时包含数字、英文字符串和特殊字符串，那么说明密码安全强度强。

在 JavaScript 脚本中，通过 search()方法调用正则表达式，完成对密码值的判断，其关键代码如下：

```
var p1 = (form.pwd.value.search(/[a-zA-Z]/)!=-1)?1:0;          //判断密码中是否包含英文字符串
var p2 = (form.pwd.value.search(/[0-9]/)!=-1)?1:0;             //判断密码中是否包含数字
```

```
var p3 = (form.pwd.value.search(/[^a-zA-Z0-9_]/)!=-1)?1:0;        //判断密码中是否包含特殊字符
var p = p1+p2+p3;                                                 //定义判断的返回值
if(p==1){
    chknew_pwd.innerHTML="<img src='images/1_03.jpg'>";           //输出密码安全强度差的图像
  }else if(p==2){
    chknew_pwd.innerHTML="<img src='images/2_03.jpg'>";           //输出密码安全强度中的图像
  }else if(p==3){
    chknew_pwd.innerHTML="<img src='images/3_03.jpg'>";           //输出密码安全强度强的图像
  }
```

实例 191　任意调整上传图片的大小

（实例位置：配套资源\SL\10\191）

实例说明

图像大小的调整对于专业的美工或者网页设计者来说不算什么，但对于程序员来说，特别是对那些不善于处理图片的程序员来说，是一个非常头疼的问题。虽说图片处理不在程序员的工作范围之内，但是，有些小问题还是自力更生的好。本实例将介绍一种方法，对图像的大小任意调整，这是解决程序员处理图片头疼的一剂良方。运行本实例，在上传图片的右侧，选择要调整图片大小的比例，然后单击下面的按钮，运行结果如图 10.21 所示。

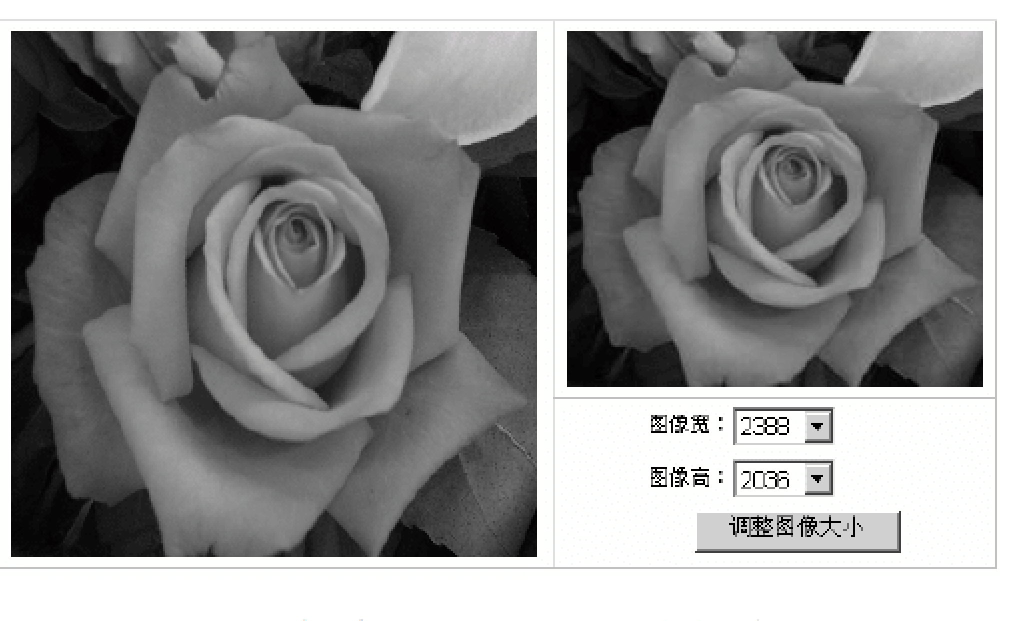


图 10.21　任意调整上传图片的大小

实现过程

具体步骤如下：

（1）创建一个图片上传功能，将图片上传到服务器指定的文件夹下，并通过文件系统函数读取出存储在指定目录下的图片。

（2）在图片输出的同时，在右侧创建一个表单，提交图片的存储位置、名称和要调整的尺寸，将数据提交到 index_ok.php 文件中。

（3）创建 index_ok.php 文件，包含 thumbnail.php 文件，实例化 image 类，调用其中的 thumb()方法，将表单中提交的数据作为参数，完成对图片大小的调整。其关键代码如下：

```
<?php
if($_POST['Submit']=="调整图像大小"){
include("thumbnail.php");                //包含图像类存储文件
$img = new image();                      //实例化图像处理类
```



```
        $return=$img->param($_POST['pic_path'])->thumb("./thumbnail/".$_POST['pic_name'], $_POST['pic_ width'], $_POST['pic_height'],1);
        echo "<script>alert('图像大小设置成功！ '); window.location.href='index.php';</script>";
        }
        ?>
```

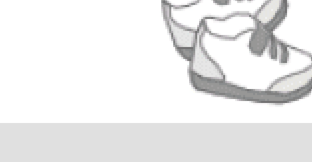


（4）创建 thumbnail.php 文件，编写 image 类，定义 thumb()方法，根据表单中提交的数据，完成图像大小的调整操作。其关键代码如下：

```
    // thumb(新图地址，宽，高，裁剪，允许放大)
    public function thumb($filename, $news_w = 100, $news_h = 100, $cut = 0, $big = 0) {
      $info = $this->getImageInfo ( $this->img );                  //获取原图信息
      if (! empty ( $info [0] )) {
            $old_w = $info [0];
            $old_h = $info [1];
            $new_w = $old_w * $news_w;
            $new_h = $old_h * $news_h;
            $type = $info ['type'];
            $ext = $info ['ext'];
            unset ( $info );
            if ($old_w < $new_h && $old_h < $new_w && ! $big) {//如果原图比缩略图小 并且不允许放大
                return false;
            }
            if ($cut == 1) {                                       //居中裁剪
                $scale1 = round ( $new_w / $new_h, 2 );
                $scale2 = round ( $old_w / $old_h, 2 );
                if ($scale1 > $scale2) {
                    $end_h = round ( $old_w / $scale1, 2 );
                    $start_h = ($old_h - $end_h) / 2;
                    $start_w = 0;
                    $end_w = $old_w;
                } else {
                    $end_w = round ( $old_h * $scale1, 2 );
                    $start_w = ($old_w - $end_w) / 2;
                    $start_h = 0;
                    $end_h = $old_h;
                }
                $width = $new_w;                                   //定义图像宽
                $height = $new_h;                                  //定义图像高
            }

            $createFun = 'ImageCreateFrom' . $type;                //新建图像
            $oldimg = $createFun ( $this->img );
            if ($type != 'gif' && function_exists ( 'imagecreatetruecolor' )) {
                $newimg = imagecreatetruecolor ( $width, $height );  //创建新的真彩图像
            } else {
                $newimg = imagecreate ( $width, $height );         //创建新的图像
            }
            if (function_exists ( "ImageCopyResampled" )) {
```





```
                    ImageCopyResampled ( $newimg, $oldimg, 0, 0, $start_w, $start_h, $width, $height,
$end_w, $end_h );                                                           //完成图像的复制
               } else {
                    ImageCopyResized ( $newimg, $oldimg, 0, 0, $start_w, $start_h, $width, $height,
$end_w, $end_h );
               }
               $type == 'jpeg' && imageinterlace ( $newimg, 1 );            //对 jpeg 图形设置隔行扫描
               $imageFun = 'image' . $type;                                 //生成图片
               ! @$imageFun ( $newimg, $filename ) && die ( '保存失败!' );
               ImageDestroy ( $newimg );                                    //销毁图像
               return $filename;                                            //返回图像名称
          }
     }
```

技术要点

任意调整上传图片大小方法实现的原理：

首先，根据表单中提交的值，获取指定图片的存储路径和名称，以及它要被调整的比例。

其次，应用 getimagesize()函数获取指定图片的数据。

再次，定义 thumb()方法，根据原始图片的数据和提交的调整比例数据，判断是根据比例缩放，还是进行裁剪。

最后，载入原始图片，并定义新的缩略图，应用 ImageCopyResampled()函数将原始图像复制到新的缩略图中，并将新的图像存储到指定位置。

（1）imagecopyresampled()函数，从原图像中采样，复制部分图像，重新生成图像。

（2）Imagecopyresized()函数，同样实现图像的复制操作。语法如下：

```
     bool imagecopyresized ( resource dst_image, resource src_image, int dst_x, int dst_y, int src_x, int
src_y, int dst_w, int dst_h, int src_w, int src_h )
```

（3）imagecopyresized()将一幅图像中的一块正方形区域复制到另一个图像中。

参数说明：

☑ dst_image：是目标图像标识符。

☑ src_image：是源图像标识符。

如果源图像和目标图像的宽度和高度不同，则会进行相应的图像收缩和拉伸。坐标指的是左上角。本函数可用来在同一幅图内部复制（如果 dst_image 和 src_image 相同的话）区域，如果区域交迭，则结果不可预知。

脚下留神：

因为调色板图像的限制（255+1 种颜色），而重新采样或过滤图像通常需要多于 255 种颜色，所以计算新的被重新采样的像素及其颜色时采用了一种近似值。对调色板图像尝试分配一个新颜色时，如果失败，则选择计算结果最接近（理论上）的颜色。这并不是视觉上最接近的颜色。这可能会产生怪异的结果，例如，空白（或者视觉上是空白）的图像。要跳过这个问题，请使用真彩色图像作为目标图像，例如，使用 imagecreatetruecolor()函数创建的图像。

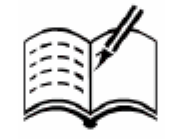


文件目录处理

本章读者可以学到如下实例：

- ⏭ 实例 192　文件操作汇总
- ⏭ 实例 193　文件类型检测
- ⏭ 实例 194　删除指定目录下的所有 ini 文件
- ⏭ 实例 195　重新定义目录的名称
- ⏭ 实例 196　获取磁盘分区的大小
- ⏭ 实例 197　遍历指定目录下的所有文件
- ⏭ 实例 198　可以屏蔽刷新功能的文本计数器
- ⏭ 实例 199　从文本文件读取注册服务条款
- ⏭ 实例 200　遍历、删除指定文件目录下的所有文件
- ⏭ 实例 201　文件属性分析
- ⏭ 实例 202　将文本文件中的内容存储到数据库中
- ⏭ 实例 203　判断文件是否被改动
- ⏭ 实例 204　目录操作汇总




Note

实例 192　文件操作汇总

（实例位置：配套资源\SL\11\192　视频位置：配套资源\SP\11\192）

实例说明

为了便于对网站的管理、维护和更新，应该设计一个能够对文件进行操作的模块，实现对文件的创建、复制、移动和删除的操作，这样能够给网站的管理工作提供很大的方便，不再因为要修改某个文件而要登录到 FTP 中，通过下载或上传实现文件的更新，这种方案能够节省很多的时间。运行本实例，实现对指定文件的创建、复制、移动和删除的操作，只要在文本框中输入要复制文件的路径和名称，在对应的文本框中输入指定文件要复制到的具体文件夹的路径和名称（包括指定文件的名称），然后单击“提交”按钮即可，运行结果如图 11.1 所示。



图 11.1　文件操作汇总

实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，设计文件汇总的页面。然后。应用 switch 语句设计一个简单的框架，完成复制、移动、创建和删除操作之间的切换。最后，编写 PHP 脚本，根据表单中提交的数据，执行不同的文件操作。其关键代码如下：

```
<?php
  if(isset($_GET['operate'])){                //获取超链接参数
        $operate=$_GET['operate'];            //定义参数变量值
  }else{
        $operate="";
  }
  switch ($operate) {                         //实现不同操作之间的转换
        case "found" :
              include ("found.php");
              break;
        case "copy" :
              include ("copy.php");
              break;
```



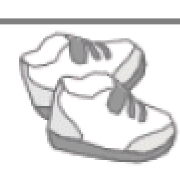
```
            case "move" :
                include ("move.php");
                break;
            case "delete" :
                include ("delete.php");
                break;
            default :
                include ("copy.php");
        }
        ?>
    <?php
    if ($_POST ['Submit'] == "复制") {                          //判断用户是否执行复制操作
        $copy = iconv ( "utf-8", "gb2312", $_POST ['copys'] );   //转换获取名称的编码格式
        $copys2 = iconv ( "utf-8", "gb2312", $_POST ['copys2'] );
        if (copy ( $copy, $copys2 )) {                          //判断是否复制成功
            echo "<script>alert('复制成功!!');</script>";          //弹出复制成功对话框
        } else {
            echo "<script>alert('复制失败!!');</script>";          //弹出复制失败对话框
        }
    }
    //省略了部分代码
    ?>
```

（2）分别创建 copy.php、delete.php、found.php 和 move.php 文件，完成不同操作的执行页面。copy.php 文件的代码如下：

```
<table width="466" height="112" border="0" cellpadding="0" cellspacing="0">
  <form name="form1" method="post" action="">
  <tr>
      <td width="111" rowspan="2" align="center">复制文件</td>
      <td width="67" align="right" valign="bottom">copy：</td>
      <td width="229" height="39" valign="bottom"><input name="copys"
          type="text" id="copys" size="22"></td>
      <td width="126" rowspan="2" align="left" valign="middle"><input
          type="submit" name="Submit" value="复制"></td>
  </tr>
  <tr>
      <td align="right" valign="top">affix：</td>
      <td height="41" valign="top"><input name="copys2" type="text"
          id="copys2" size="22"></td>
  </tr>
  <tr>
      <td colspan="4" align="center">例如：F:\xampp\htdocs\MR\05\012\test.txt</td>
  </tr>
  </form>
</table>
```

技术要点

本实例主要应用 fopen()、copy()、rename()和 unlink()4 个函数完成文件的创建、复制、

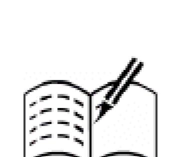


移动和删除操作。

（1）fopen()函数。

fopen()函数用于打开文件或者URL。函数的语法如下：

```
int fopen(string filename, string mode);
```



Note

创建文件主要应用fopen()函数的mode参数，mode参数的可选值如下：

- ☑ 'w' 写入方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在，则进行创建。
- ☑ 'w+' 读写方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在，则进行创建。
- ☑ 'a' 写入方式打开，将文件指针指向文件末尾。如果文件不存在，则进行创建。
- ☑ 'a+' 读写方式打开，将文件指针指向文件末尾。如果文件不存在，则进行创建。

（2）copy()函数。

copy()函数主要用于复制文件。函数的语法如下：

```
int copy(string source, string dest);
```

将文件从source复制到dest。如果成功则返回true，否则返回false。

（3）rename()函数。

rename()函数主要用于文件重命名。函数的语法如下：

```
bool rename(string oldname,newname);
```

本函数将oldname重命名为newname，成功则返回true，否则返回false。

（4）unlink()函数。

unlink()函数主要用于删除文件。函数的语法如下：

```
bool unlink(string filename);
```

本函数用于删除文件。如果文件删除成功则返回true，否则返回false。

多学两招：

在应用文件操作函数执行文件的创建、移动、复制和删除操作时，由于实例的页面编码格式是UTF-8，所以，从表单中获取的元素值，不能够直接应用到文件操作函数中，必须对元素值进行编码转换，由UTF-8编码转换为GB2312编码，否则文件操作函数不能够正确的执行。

实例193　文件类型检测

（实例位置：配套资源\SL\11\193）

实例说明

在开发程序的过程中，很多时候都需要获取到文件的后缀名，并根据后缀名作出一些





判断。那么如何才能获取到文件的后缀名称呢？在本实例中将对这个问题进行解答。运行本实例，将检测表单中提交文件的后缀，并根据后缀名称，作出相应的判断。其运行结果如图 11.2 所示。

图 11.2　文件类型检测

实现过程

具体步骤如下：

（1）创建名称为 index 的 PHP 文件。

（2）添加表单，设置文件域、提交按钮，使用 POST 方法，设置 enctype="multipart/form-data"，将数据提交到本页。

（3）通过$_FILES 获取上传文件的相关信息。

（4）应用 is_dir()函数判断指定的服务器文件夹是否存在，如果不存在则应用 mkdir()函数创建文件夹。

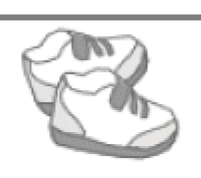
（5）应用 is_uploaded_file()函数判断文件是否是 HTTP POST 上传。

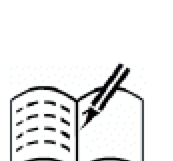
（6）获取上传文件名称的后缀，根据后缀的不同，定义不同的存储路径。

（7）应用 move_uploaded_file()函数执行文件上传的操作。

其中 index.php 文件的关键代码如下：

```
<?php
if (! empty ( $_FILES ['up_picture'] ['name'] )) {                    //判断上传内容是否为空
  if ($_FILES ['up_picture'] ['error'] > 0) {                         //判断文件是否可以上传到服务器
        echo "上传错误:";
        switch ($_FILES ['up_picture'] ['error']) {                   //判断上传错误代码
              case 1 :                                                //根据错误代码输出错误提示
                    echo "上传文件大小超出配置文件规定值";
                    break;
              case 2 :
                    echo "上传文件大小超出表单中约定值";
                    break;
              case 3 :
                    echo "上传文件不全";
                    break;
              case 4 :
                    echo "没有上传文件";
                    break;
        }
  }
} else {
  if (! is_dir ( "./txt/" )) {                                        //判断指定目录是否存在
        mkdir ( "./txt/" );                                           //创建目录
  }
  if (! is_dir ( "./pic/" )) {                                        //判断指定目录是否存在
        mkdir ( "./pic/" );                                           //创建目录
  }
  if (! is_dir ( "./fla/" )) {                                        //判断指定目录是否存在
```





```
            mkdir ( "./fla/" );                                    //创建目录
        }
    }
    if(!empty($_POST["submitvalue"])){
    if (is_uploaded_file ( $_FILES ['up_picture'] ['tmp_name'] )) {    //判断文件是否是 HTPP POST 上传
      $type = $_FILES ['up_picture'] ['name'];                     //获取上传文件的名称
      $types = strtolower ( strstr ( $type, '.' ) );               //获取上传文件的后缀
      if ($types == ".txt" || $styles == ".doc") {
            $path = './txt/' . time () . strstr ( $_FILES ['up_picture'] ['name'], '.' ); //定义上传文件名称和存
储位置
      } elseif ($types == ".jpg" || $styles == ".gif" || $styles == ".bmp") {
            $path = './pic/' . time () . strstr ( $_FILES ['up_picture'] ['name'], '.' ); //定义上传文件名称和存
储位置      } else {
    //定义上传文件名称和存储位置
            $path = './fla/' . time () . strstr ( $_FILES ['up_picture'] ['name'], '.' );
      }
      if (! move_uploaded_file ( $_FILES ['up_picture'] ['tmp_name'], $path )) {      //执行上传操作
            echo "上传失败！ ";
      } else {
            echo "文件类型：" . $types . " 上传成功，大小为：" . $_FILES ['up_picture'] ['size'];
      }
    } else {
    echo "上传文件：" . $_FILES ['up_pictute'] ['name'] . "不合法！ ";
    }
    }
    ?>
```

脚下留神：

处理页为了判断用户是否已经执行上传操作，在<form>表单中添加一个名称为 submitvalue 的隐藏域。并通过 if(!empty($_POST["submitvalue"])) 语句判断传递的 submitvalue 值是否存在。如果该传递值为空则没有执行上传操作，反之则执行了。如不添加该语句则有可能会导致系统报错。

技术要点

在本实例中应用 move_uploaded_file()函数实现文件上传的操作，并根据文件的后缀进行判断，将不同类型的文件存储在不同的服务器文件夹下。

获取上传文件的后缀，首先，应用$_FILES 全局变量获取上传文件的名称。然后，应用 strstr()函数对上传文件的名称进行截取，截取字符串中“.”后的所有字符串。最后，应用 strtolower()函数将字符串转换成小写。

strstr()函数，获取一个指定字符串在另一个字符串中首次出现的位置到后者末尾的子字符串。语法如下：

```
string strstr ( string haystack, string needle)
```

参数说明：

☑ haystack：必选参数，指定从哪个字符串中进行搜索。



☑ needle：必选参数，指定搜索的对象。如果该参数是一个数值，那么将搜索与这个数值的 ASCII 值相匹配的字符。

如果执行成功，则返回剩余字符串（存在相匹配的字符）；如果没有找到相匹配的字符，则返回 false。

多学两招：

文件后缀的获取可以应用到很多地方，最为常用的就是通过它来控制上传文件的类型；还有在文件的读取中，判断哪些文件可以读取，哪些不可以读取。

实例 194　删除指定目录下的所有 ini 文件

（实例位置：配套资源\SL\11\194　视频位置：配套资源\SP\11\194）

实例说明

在文件操作汇总中已介绍过删除指定目录下所有文件的操作，在本实例中，将完成一个更加具体的操作，删除指定目录下的所有 ini 文件。其运行结果如图 11.3 所示。



图 11.3　删除指定目录下的所有 ini 文件

实现过程

具体步骤如下：

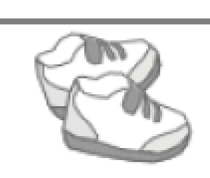
（1）本实例的创建过程与遍历、删除指定目录下的所有文件是基本相同的，在 index.php 文件中的不同之处已经在关键技术中给出，这里不再赘述。

（2）创建 look_file.php 文件，将获取文件名称的编码格式进行转换，然后循环读取文件中的信息，最后将特殊字符转换后的内容输出，完成对“.INI”后缀文件的读取操作。其代码如下：

```
<?php
$catalog = iconv ( "utf-8", "gb2312", urldecode ( $_GET ['catalog'] ) );
$filename = iconv ( "utf-8", "gb2312", urldecode ( $_GET ['filename'] ) ); $type = iconv ( "utf-8",
"gb2312", urldecode ( $_GET ['type'] ) );      //获取文件的信息，设置编码
$arr = file ( $catalog . "\\" . $filename );                    //读取文件
foreach ( $arr as $value ) {                                    //循环输出文件内容
  $value = htmlentities ( $value, ENT_COMPAT, "UTF-8" );        //特殊字符的转换
  echo $value . "<br>";                                         //输出内容
}
?>
```

（3）创建 delete.php 文件，当用户在单击首页列表中文件名所对应的“删除”超链接，将完成对指定文件的删除操作。

Note



技术要点



Note

有关目录、文件的删除方法已经在前面实例文件操作汇总中讲解过，这里就不再赘述。而要删除指定目录下的特定格式文件，其实现的原理与删除所有文件相同，只是在创建删除链接前需要对文件的格式进行判断，如果文件的格式是“.INI”类型，那么就输出删除的超链接，否则将直接输出文件的名称。其关键代码如下：

```
$catalog = getcwd () . "\\$gain_directory";                    //子目录
$ext = substr ( $gain_directory, strrpos ( $gain_directory, "." ) );    //获取文件的后缀
//判断如果文件的后缀是.ini，哪么则创建删除超链接
if (strtoupper ( $ext ) == ".INI") {                    //判断获取文件名称后缀是否为“.INI”
  echo "<a href='delete.php?catalog=".urlencode($catalog)."&filename=" . urlencode(getcwd ()) . "'
title='删除目录或者文件' >删除</a>";
} else {                    //如果文件的后缀不是.ini，则直接输出文件名称
  echo iconv ( "gb2312", "utf-8", $gain_directory );        //输出转换编码格式后的文件名
  }
```

多学两招：

在进行程序开发的过程中，要学会合理地运用自己的编程资源。开发的功能模块要具有重用价值，一个好的功能模块，不只是在一个程序中可以使用。诸如数据库的连接、操作方法以及分页技术一次开发后，将会在很多的程序中使用，避免对类似的程序进行重复开发，浪费不必要的时间。

作为一个程序员，不只要每次在开发时都考虑新的功能、新的技术，更重要的是充分运用已有资源，既可以提高程序的开发效率，又可以避免开发新技术产生的一些问题。

实例 195　重新定义目录的名称

（实例位置：配套资源\SL\11\195）

实例说明

在对网站进行管理和维护的过程中，经常会修改文件夹的名称，它也是目录的一项基本操作。本实例中将介绍更新目录名称的方法。运行本实例，单击当前目录中文件夹后的“重命名”超链接，将进入到如图 11.4 所示的页面，在这个页面中完成对指定目录的重命名。

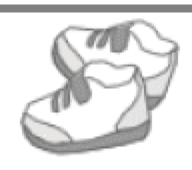


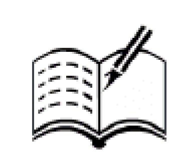
图 11.4　重新定义目录的名称

实现过程

具体步骤如下：

（1）创建 index.php 文件。输出当前目录的文件和目录信息，并且为输出的目录创建



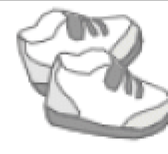


“重命名”的超链接，其传递的参数是指定目录的存储位置和当前目录的位置。其关键代码如下：

```
<table width="650" border="0" cellspacing="0" cellpadding="0">
        <tr>
            <td width="50" height="23"> </td>
            <td width="450"><span class="STYLE1">当前目录：<?php echo getcwd ();?></span></td>
        </tr>
      <?php
            $lookdir = getcwd ();                              //获得当前脚本目录
            $i = 0;
            if (is_dir ( $lookdir )) {                         //检测是否是合法目录
                if ($opendir = opendir ( $lookdir )) {         //打开目录
                    while ( $li = readdir ( $opendir ) ) {     //读取目录
        ?>
        <tr>
            <td height="23"> </td>
            <td>
        <?php
            $i ++;                                             //执行变量$i 自增操作
            echo " $i: " . iconv ( "gb2312", "utf-8", $li );   //转换编码格式
        ?>
    </td>
            <td>
    <?php
            if ($li == "." || $li == "..") {                   //判断变量$li 是否为“.”或者“..”
                echo "  ";
            } elseif (filetype ( $li ) == "file") {            //否则判断变量$li 是否为文件
                echo "  ";
            } else {
                $cata = getcwd () . "\\" . iconv ( "gb2312", "utf-8", $li );
        ?>
        <a href="rename.php?catalog=<?php echo urlencode ( $cata ); ?>&dir_name=<?php echo
urlencode ( getcwd () );?>"title="更新目录名称" target="_blank">重命名 </a>
        <?php    }    ?>
    </td>
        </tr>
    <?php    }    }    }    ?>
    </table>
```

（2）创建 rename.php 文件，根据超链接传递的参数值，创建 form 表单，完成目录的重新命名提交，并且在 rename.php 文件中完成重命名的操作。其关键代码如下：

```
    <?php
    if (isset($_POST['Submit'])) {                                         //判断用户是否执行提交操作
    if ($_POST['Submit'] == "提交") {
        $old_name=iconv("utf-8","gb2312",$_POST['old_name']);              //获取原始目录名称
        $new_name=iconv("utf-8","gb2312",$_POST['new_name']);              //获取新目录名称
        if (file_exists ( $old_name )) {
            if (rename ( $old_name, $new_name )) {                         //把原目录重新命名
                echo "<script>alert('目录重命名成功！'); window.location.href='index.php';</script>";
```

```
                }else{
                    echo "<script>alert('目录重命名失败！'); window.location.href='rename.php';</script>";
                }
            } else {
                echo "<script>alert('目录不存在！ '); window.location.href='index.php';</script>";
            }
        }
    }
?>
```

链接的参数向下一个页面中传递值时，最佳的方法就是应用 urlencode()函数对参数值进行编码，其最大的优点就是避免传递中文字符串或者空格时出现乱码，同时还隐藏了传递的参数值，有一定的保护作用。

技术要点

目录的重命名同样应用的是 rename()函数，其语法结构如下：

```
bool rename ( string oldname, string newname [, resource context] )
```

该函数尝试把 oldname 重命名为 newname。如果成功则返回 true，失败则返回 false。

多学两招：

在通过超链接的参数向下一个页面中传递值时，最佳的方法就是应用 urlencode()函数对参数值进行编码，其最大的优点就是避免传递中文字符串或者空格时出现乱码，同时还隐藏了传递的参数值，有一定的保护作用。

实例 196　获取磁盘分区的大小

（实例位置：配套资源\SL\11\196）

实例说明

通过文件系统函数不但可以对目录、文件进行操作，获取目录文件的相关信息，而且可以获取到磁盘分区的大小。运行本实例，将根据文本框提交的目录，获取到该目录所在磁盘分区的大小，以及该目录下的所有文件。其运行结果如图 11.5 所示。



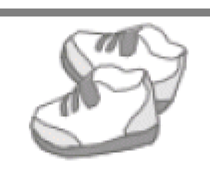
图 11.5　获取磁盘分区的大小

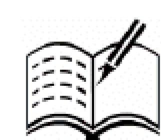
实现过程

具体步骤如下：

（1）创建 index.php 文件。

（2）添加表单，设置文本框、提交





按钮，使用 POST 方法，将数据提交到本页。

（3）通过$_POST 获取表单提交的目录路径，首先，判断获取的目录是否合理。然后，通过 iconv()函数对获取的字符串进行编码转换。接着，对获取的字符串进行截取，获取该目录所在的磁盘分区，并应用 disk_total_space()函数获取磁盘分区的大小。最后，应用 opendir()和 readdir()函数读取提交目录下的内容。

index.php 的关键代码如下：

```
<?php
if (isset($_POST ['Submit'])){                              //判断用户是否执行提交操作
if ($_POST ['file_name'] != "" && is_file ( $_POST ['file_name'] ) == false) {
    $file_name = iconv ( "utf-8", "gb2312", $_POST ['file_name'] );     //编码格式转换
    if (file_exists ( $file_name )) {                               //判断目录是否存在
        $len = stripos ( $file_name, ":" );                         //截取字符串
        $dir = substr ( $file_name, 0, $len + 1 );          //获取提交目录所在磁盘
        $filesize_z=disk_total_space ( $dir );                      //获取目录总大小
        $filesize_z=number_format($filesize_z/(1024*1024*1024),2,".",""); //数字的格式化
        $filesize_s=disk_free_space ( $dir );                       //获取磁盘剩余空间
        $filesize_s=number_format($filesize_s/(1024*1024*1024),2,".",""); //字节的格式化
        echo "本地磁盘（" . $dir . "） 总大小：" . $filesize_z . " GB   可 用 空 间：
".$filesize_s." GB <br><br>";
        echo $_POST ['file_name'] . " 目录下的内容：" . "<br>";
        $i = 0;
        $list = opendir ( $file_name );                                //打开目录
        while ( $read_list = readdir ( $list ) ) {                     //读取目录
            $i ++;
            echo " $i: " . iconv ( "gb2312", "utf-8", $read_list ) . " <br> "; //输出目录中的内容
        }
        closedir ( $list );                                        //关闭目录
    } else {
        echo "<script>alert('目录不存在！');</script>";             //提示用户目录不存在
    }
 } else {
    echo "<script>alert('请输入正确的目录路径！');</script>";     //提示用户输入正确路径
 }
}
?>
```

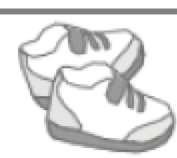
技术要点

获取磁盘分区的大小应用的是 disk_total_space()函数；获取磁盘分区的剩余空间应用的是 disk_free_space()函数。

disk_total_space()函数，获取一个目录的磁盘总大小，语法如下：

```
float disk_total_space ( string directory )
```

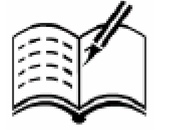
该函数根据参数 directory 提供的一个目录字符串，返回相应的文件系统或磁盘分区的所有字节数。



disk_free_space()函数，获取一个目录的可用空间，语法如下：

```
float disk_free_space(string directory)
```

参数 directory 用于指定文件系统或磁盘分区。



多学两招：

strripos()函数，判断指定字符串在另一字符串中最后出现的位置。

通过本函数可以获取到指定字符串 A 在另一字符串 B 中最后出现的位置，其返回值为 int 型，根据这个返回值就可以对字符串 B 以字符串 A 为分隔点进行截取，在执行截取的操作时需要应用到 substr()函数。

实例 197　遍历指定目录下的所有文件

（实例位置：配套资源\SL\11\197）

实例说明

在网站的后台管理系统中，经常需要对网站服务器中的文件进行管理和维护，有时需要添加一个文件夹或删除某个文件夹或者文件，为了更好地查看到这些文件或者文件夹，最好的方法就是创建一个文件查询系统，通过它可以查看到指定文件夹下的所有文件。运行本实例，在文本框中输入一个指定的文件夹，单击“提交”按钮，如果该文件夹存在，就可以显示出该文件夹下包括的所有文件，运行结果如图 11.6 所示。

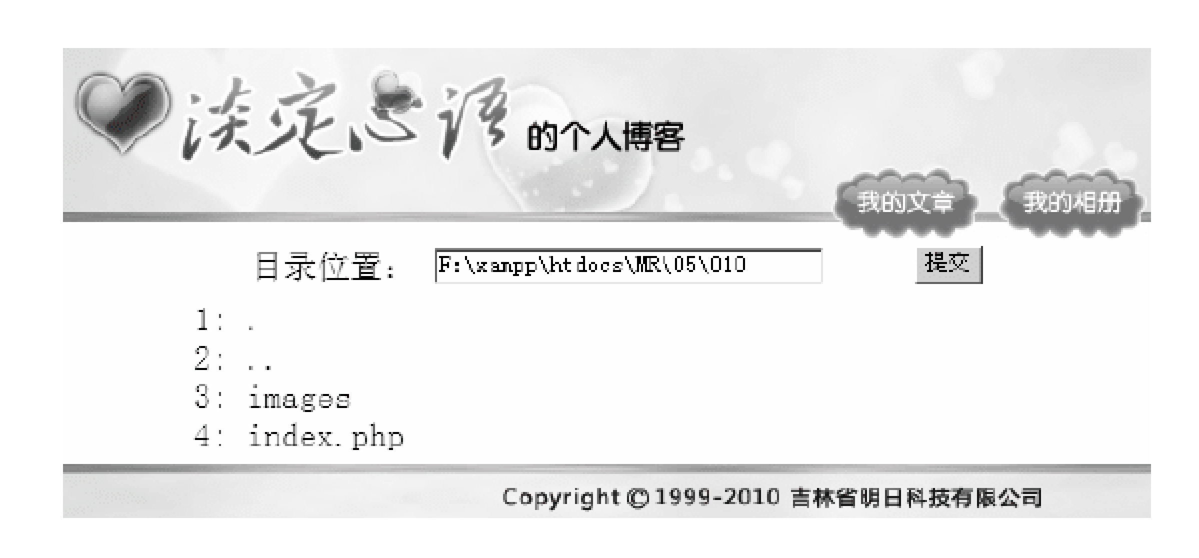


图 11.6　遍历指定目录下的所有文件

实现过程

具体步骤如下：

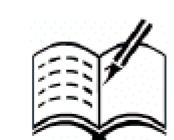
（1）创建 index.php 文件。添加一个表单，定义表单元素，通过 POST 方式提交目录的路径。

（2）在 index.php 中，判断提交的目录路径是否为空，如果不为空，则打开目录，读取目录中的文件，并输出读取到的文件，关闭目录；否则，输出目录不存在。其关键代码如下：

```
<?php
if(isset($_POST['Submit'])){                                    //判断用户是否执行提交操作
if(!file_exists ( $look_file )){                                //检测指定的目录是否存在
        print $look_file."目录不存在!" ;
}else{
        $i = 0;
        if( is_dir ( $look_file )){                             //检测是否是合法目录
        if ($list = opendir ( $look_file )){                    //打开目录
```



```
                    while ($read_list = readdir( $list )){                    //读取目录
                    $i++ ;
                    echo " $i: ".iconv("gb2312","utf-8",$read_list)." <br> " ; //输出目录中的内容
                    }
                }
            }
        closedir ( $list ) ;                                               //关闭目录
    }
    }
    ?>
```



脚下留神：

bool is_dir()函数结果会被含纯；该函数不能作用于远程文件，被检查的文件必须通过服务器的文件系统访问。

技术要点

本实例实现的关键是目录和文件处理函数的应用，包括 file_exists()、is_dir()、opendir()、readdir()和 closedir()函数。

（1）file_exists()：检查文件或目录是否存在。语法如下：

```
bool file_exists(string filename)
```

如果由 filename 指定的文件或目录存在则返回 true，否则返回 false。

（2）is_dir()：判断给定的文件名是否是一个目录。

```
bool is_dir(string filename)
```

如果文件名存在并且为目录则返回 true。如果 filename 是一个相对路径，则按照当前工作目录检查其相对路径。

（3）opendir()：打开目录句柄。语法如下：

```
resource opendir ( string path)
```

本函数返回一个目录句柄，可以在 closedir()，readdir()之前调用。如果 path 不是一个合法的目录或者因为权限限制或文件系统错误而不能打开目录，opendir()返回 false 并产生一个错误信息。

（4）readdir()：从目录句柄中读取条目。语法如下：

```
string readdir( resource dir_handle)
```

本函数返回目录中下一个文件的文件名。文件名以在文件系统中的排序返回。

（5）closedir()：关闭目录句柄。语法如下：

```
string closedir( resource dir_handle)
```

本函数关闭由 dir_handle 指定的目录流，目录流必须被 opendir()打开过。

多学两招：

本实例中只是可以浏览到指定目录下包含的文件，不可以对文件进行操作。





实例 198　可以屏蔽刷新功能的文本计数器

（实例位置：配套资源\SL\11\198　视频位置：配套资源\SP\11\198）

实例说明

网站的计数器对于网站管理者来说是一个非常值得关注的部分，它记录了网站被访问的次数，客观地反映了网站受欢迎的程度。

而文本计数器是最简单的一种，采用将数据存储于文本文件中。在本实例中，将介绍这种文本计数器的实现方法，并且将重点阐述如何屏蔽网页刷新对计数器的影响。运行本实例，将输出如图 11.7 所示界面。

最新动态　· 此时是：2010年7月6日16:33:51　您是本网站第1800008 位访客！

图 11.7　可以屏蔽刷新功能的文本计数器

实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，设计网页页面。然后，初始化 SESSION 变量。接着，编写 PHP 脚本，通过文件系统函数完成网站访问量的统计，通过 SESSION 屏蔽网页刷新对计数器的影响。最后，输出网站的访问量和当前时间。其关键代码如下：

```
<?php
session_start ();                    //初始化一个 SESSION 变量
?>
<?php
//使用文本存储数据
if ($_SESSION [temp] == "") {        //判断$_SESSION[temp]==""的值是否为空，其中的 temp 为自定义的变量
  if (($fp = fopen ( "counter.txt", "r" )) == false) {
        echo "打开文件失败!";
  } else {
        $counter = fgets ( $fp, 1024 );      //读取文件中数据
        fclose ( $fp );                      //关闭文本文件
        $counter ++;                         //计数器增加 1
        $fp = fopen ( "counter.txt", "w" );  //以写的方式打开文本文件<!---->
        fputs ( $fp, $counter );             //将新的统计数据增加 1
        fclose ( $fp );                      //关闭文件
  }
  $_SESSION [temp] = 1;                      //计数器的值增加后,为$_SESSION[temp]赋值 1
}
//从文本文件中读取统计数据
if (($fp = fopen ( "counter.txt", "r" )) == false) {
        echo "打开文件失败!";
} else {
        $counter = fgets ( $fp, 1024 );      //读取文本文件的数据
```





```
        fclose ( $fp );
    }
?>
```

（2）在实例根目录下创建 counter.txt 文本文件，用于存储网站的访问量。

技术要点

文本计数器的设计思路如下：

首先，判断文本文件是否存在，如不存在则打开失败；如打开成功则继续执行并读取文件中的数据，将计数器增加 1；

其次，新用户访问网页时，处理页以写入的方式重新打开文件，把新的统计数据写入到文件中，关闭文件；

最后，重新打开文件，读取并输出文件中的数据。

其操作流程如图 11.8 所示。

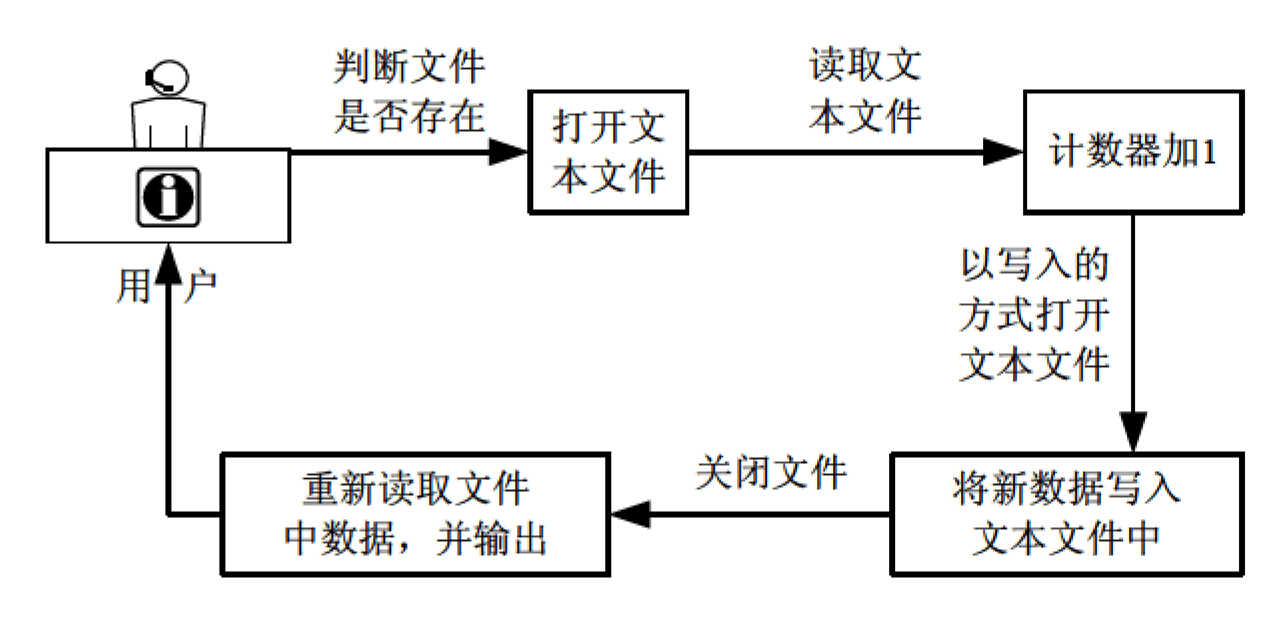


图 11.8　文本计数器的操作流程

按照这个原理设计的计数器，当页面刷新时计数器的值就会增加，那么这个计数器就没有任何意义。所以要想这个计数器有意义，必须屏蔽页面刷新对计数器的影响。

这里通过 SESSION 来实现这个功能，其原理如图 11.9 所示。

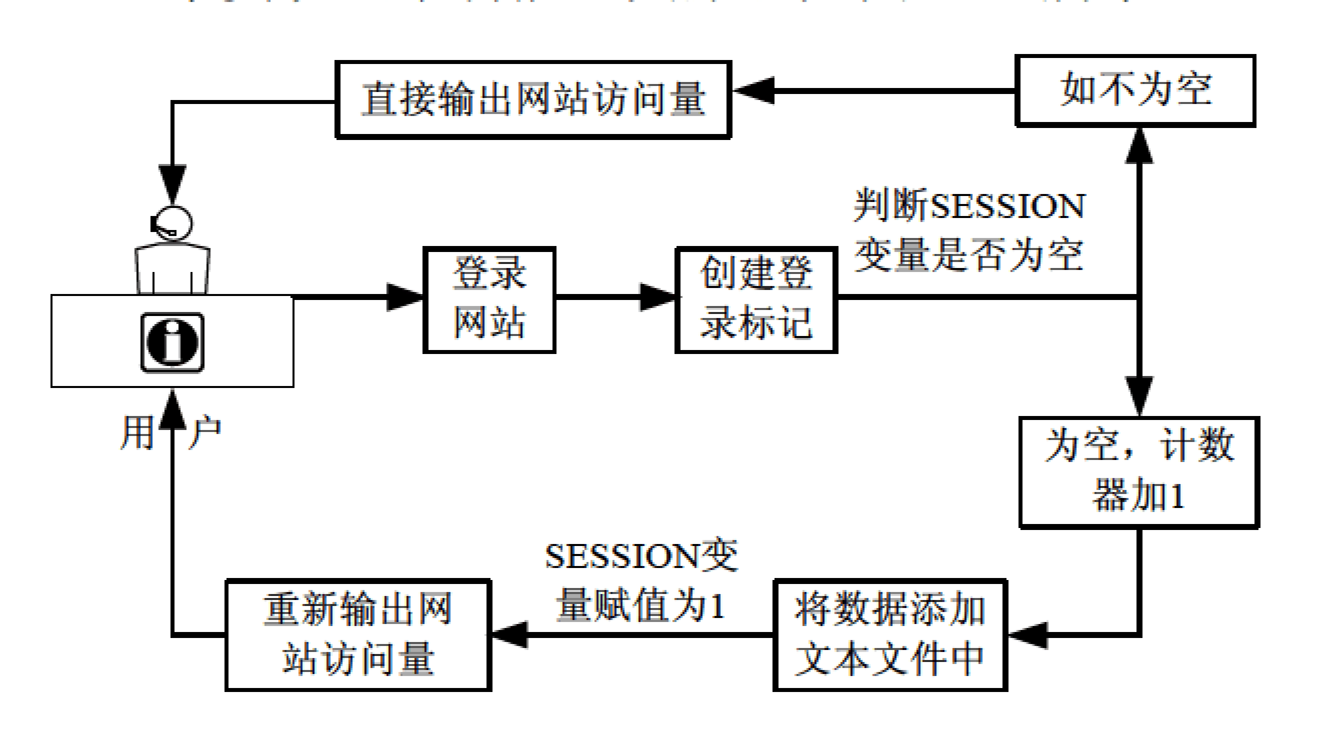
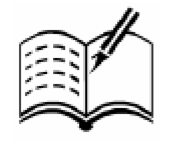


图 11.9　网站计数器的设计原理

多学两招：

首先，在网页被访问时，初始化一个 SESSION 变量，并赋给其一个空值。然后，判断 SESSION 变量的值是否为空，如果为空，则将计数器的值增加 1，并且为 SESSION 变量赋值为 1。此时，在当前页中，SESSION 变量的值已经不为空，无论如何刷新，SESSION 变量的值都不会改变，所以计数器的值也不会增加。


Note

实例 199　从文本文件读取注册服务条款

（实例位置：配套资源\SL\11\199　视频位置：配套资源\SP\11\199）

实例说明

在网站开发的过程中，经常会创建注册服务条款或者会员须知之类的文件。处理此类文件的最直接方法是将它生成一个独立的页面，而最实用的方法是将它存储于独立的文本文件中，从文本文件中读取这些服务条款，它不占用数据库的空间，而且不占用过多的页面。运行本实例，实现一个用户注册的功能，并从文本文件中读取服务条款，运行结果如图 11.10 所示。



图 11.10　从文本文件中读取注册服务条款

实现过程

具体步骤如下：

（1）创建 index.php 文件。

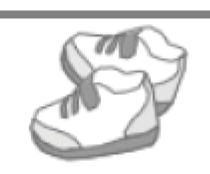
首先，设计用户注册页面。

其次，添加用户注册的表单及表单元素，将表单元素的值提交到 index_ok.php 页。

再次，编辑 JavaScript 脚本，通过 chkreginfo()方法对表单元素的值进行判断。

最后，在文本域中编写 PHP 脚本，通过 file()函数读取指定文本文件中的数据，通过 foreach 语句循环输出注册服务条款。其关键代码如下：

```
<textarea name="register" cols="40" rows="5" id="register">
<?php
$arr = file ( 'files/register.txt' );                    //读取 register.txt 文本文件中的内容
foreach ( $arr as $value ) {                             //循环输出文本文件中的内容
echo iconv ( "gb2312", "utf-8", $value );
}
?>
</textarea>
```



（2）创建 index_ok.php 文件，通过$_POST[]方法获取表单中提交的数据，输出用户的注册信息。

（3）在根目录下创建 js 脚本文件夹，编写 check.js 脚本文件，完成对表单元素值的判断。



技术要点

从文本文件中读取注册服务器条款，可以分为两个部分：第一部分是表单中文件域的创建；第二部分是应用文件系统函数读取、输出指定文本文件中的数据。

这里应用文件系统中的 file()函数读取文本文件中的数据。

file()函数将整个文件的内容读入到一个数组中。成功返回数组，数组中的每个元素都是文件中对应的一行，包括换行符在内；失败返回 false。语法如下：

```
array file ( string filename [, int use_include_path [, resource context]] )
```

其参数与 readfile()函数相同，唯一区别是该函数返回值是数组。

多学两招：

在文件系统函数中，有多个可以读取文本文件中数据的函数，为了让读取到的数据输出时条理更加清晰，建议使用 file()函数，因为该函数将读取到的数据写入一个数组中，数组中的每个元素是文件中对应的一行，这样在通过 foreah 语句输出数组中元素时，就是按照原来文件中的行进行输出，确保了内容输出的条理性。

实例 200　遍历、删除指定文件目录下的所有文件

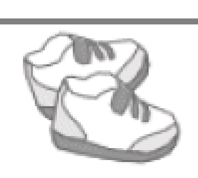
（实例位置：配套资源\SL\11\200）

实例说明

通过对网站目录的遍历，能够更快地了解网站的结构、网站文件的存储位置；通过对文件内容的遍历，掌握网站中每个文件的作用、同时也便于对目录和文件的管理。本实例中将介绍如何实现目录和文件的遍历以及删除。运行本实例，如图 11.11 所示。



图 11.11　遍历、操作指定目录下的所有文件



实现过程



Note

具体步骤如下：

（1）创建 index.php 文件。应用目录、文件处理函数和字符串处理函数完成对指定目录下文件的读取操作，并设置查看文件内容的超链接和删除指定文件的超链接。在设置超链接的参数时，应用 urlencode()函数对字符串进行编码，其完整代码请参考本书配套资源。

（2）创建 look_file.php 文件，完成对指定文件内容的读取操作。其关键代码如下：

```
<?php
$catalog = iconv ( "utf-8", "gb2312", urldecode ( $_GET [catalog] ) );  //获取文件的信息，设置编码
$filename = iconv ( "utf-8", "gb2312", urldecode ( $_GET [filename] ) ); //获取文件的信息，设置编码
$type = iconv ( "utf-8", "gb2312", urldecode ( $_GET [type] ) );         //获取文件的信息，设置编码
$types = strtoupper ( $type );                                           //完成字符串的大小写转换
if ($types == ".PHP") {                                                  //显示 PHP 文件的内容
  $arr = file ( $catalog . "\\" . $filename );                           //读取文件
  foreach ( $arr as $value ) {                                           //循环输出文件内容
        $value = htmlentities ( $value, ENT_COMPAT, "UTF-8" );           //特殊字符的转换
        echo $value . "<br>";                                            //输出内容
  }
} else {
  $fp = fopen ( $catalog . "\\" . $filename, "r" );                      //显示 txt 文件的内容
  while ( $line = fgets ( $fp ) ) {
        $line = htmlentities ( $line, ENT_COMPAT, "GB2312" );            //特殊字符的转换
        echo iconv ( "gb2312", "utf-8", $line );                         //输出文件内容
  }
  fclose ( $fp );
}
?>
```

（3）创建 delete.php 文件，用户单击首页文件对应的“删除”超链接，完成对指定文件的删除操作。

技术要点

遍历、操作指定目录下的所有文件关键是目录、文件处理函数和字符串处理函数的综合运用。

关键的目录处理函数如下：

（1）chdir()函数，改变当前目录。语法如下：

```
bool chdir(string directory)
```

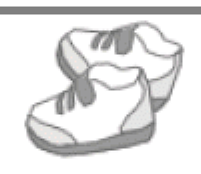
参数 directory 用于指定更改后目录的位置。操作成功返回 true，否则返回 false。

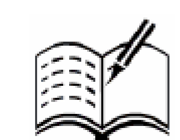
（2）getcwd()函数，获取当前工作的目录。语法如下：

```
string getcwd( void);
```

（3）rmdir()函数，删除目录。语法如下：

```
bool rmdir ( string dirname )
```





尝试删除参数 dirname 所指定的目录。该目录必须是空的，而且要有相应的权限。如果成功则返回 true，失败则返回 false。

在对文件进行操作时，主要应用到如下几个函数：

（1）fopen()函数、fgets()函数、readfile()函数和 fclose()函数，有关这 4 个函数已经在前面实例文件操作汇总中介绍过，这里不再赘述。

（2）filectime()函数，返回指定文件的索引节点修改时间。语法如下：

```
int filectime(string filename);
```

返回文件上次索引节点被修改的时间，如果出错则返回 false。时间以 UNIX 时间戳的方式返回。

（3）filemtime()函数，返回指定文件 filename 的最后修改时间。语法如下：

```
int filemtime(string filename);
```

返回文件上次被修改的时间，失败则返回 false。时间以 UNIX 时间戳的方式返回。

（4）unlink()函数，删除文件。语法如下：

```
bool unlink ( string filename )
```

删除由字符串 filename 指定的文件。成功则返回 true，失败则返回 false。

在完成对目录和文件的遍历以及删除过程中，应用到下面的字符串函数：

substr()函数，从指定的字符串中按照指定的位置截取一定长度的字符。语法如下：

```
string substr(string str,int start,int length)
```

参数说明：

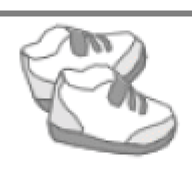
- ☑ str：指定字符串对象。
- ☑ start：指定开始截取字符串的位置。如果参数 start 为负数，则从字符串的末尾开始截取。
- ☑ length：可选参数，指定截取字符的个数，如果 length 为负数，则表示取到倒数第 length 个字符。

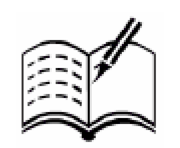
指点迷津：

substr()函数中参数 start 的指定位置是从 0 开始计算的，即字符串中的第一个字符表示 0。

多学两招：

在通过 URL 传递字符串参数时，尽量应用 urlencode()函数对传递的参数值进行编码，这样不但可以保证传递参数值的安全，而且可以防止传递的参数值出现乱码的问题。例如，本实例中在执行指定目录、文件的删除操作时，在执行成功跳转到上级页面时，如果不对 URL 传递的字符串参数值进行编码，那么就不能够完成跳转的操作，会弹出一个错误提示。





Note

实例201 文件属性分析

（实例位置：配套资源\SL\11\201）

实例说明

如果说判断目录、文件是否存在是对目录、文件进行操作的前提条件，那么获取文件属性则是对文件进行操作的必要条件。因为在执行一些特殊的操作之前，必须对文件的类型、大小或者修改时间进行判断。在本实例中，将介绍如何获取文件的类型、大小和修改时间。运行本实例，在文本框中输入一个正确目录或者文件的路径，单击“提交”按钮，运行结果如图11.12所示。



图11.12　文件属性分析

实现过程

具体步骤如下：

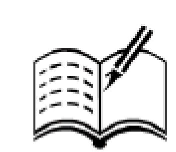
（1）创建index.php文件，设计页面布局。

（2）添加表单，设置目录、文件提交的文本框，提交目录或者文件的路径，并设置提交按钮。

（3）创建PHP脚本，获取表单中提交的元素值。首先，判断提交的值是否为空，如果不为空，则应用iconv()函数对字符串的编码格式进行转换。然后，应用file_exists()函数判断指定的目录或者文件是否存在，如果存在则获取目录或者文件的类型、大小以及修改时间。其关键代码如下：

```
<?php
if (isset($_POST ['file_name'])) {                                      //判断表单提交的值是否为空
  $file_name = iconv ( "utf-8", "gb2312", $_POST ['file_name'] );        //完成编码格式的转换
  if (file_exists ( $file_name )) {                                     //判断目录或者文件是否存在
        $file_type = filetype ( $file_name );                           //获取文件类型
        echo "<br>文件类型：" . $file_type . "<br>";
        if ($file_type != "dir") {                                      //判断如果不是目录
              $file_size = filesize ( $file_name );                     //获取文件的大小
              echo "文件大小：" . $file_size . " 字节" . "<br>";
        }
        $file_mtime = filemtime ( $file_name );                  //获取目录或者文件的修改时间
        echo "修改时间：" . date ( "Y-m-d H:i:s", $file_mtime ); //对返回的时间戳进行格式化输出
} else {
        echo "<script>alert('目录、文件不存在！');</script>";
}
} else {
echo "<script>alert('请输入正确的目录、文件路径！');</script>";
}
?>
```





Note

技术要点

PHP 中获取文件类型是 filetype()函数，获取文件大小是 filesize()函数，而获取修改时间是 filemtime()函数。

filetype()函数，获取文件类型。其语法如下：

```
string filetype ( string filename )
```

该函数返回文件的类型。类型值包括：fifo、char、dir、block、link、file 和 unknown。

如果出错则返回 false。如果 stat 调用失败或者文件类型未知的话 filetype()还会产生一个 E_NOTICE 级别的错误消息。

filesize()函数，获取文件大小。

```
int filesize ( string filename )
```

该函数返回文件大小的字节数，如果出错返回 false（在错误报告级别为 E_WARNING 的情况下）。

指点迷津：

因为 PHP 的整数类型是有符号的，并且大多数平台使用 32 位整数，filesize()函数在碰到大于 2GB 的文件时可能会返回非预期的结果。对于 2～4GB 之间的文件通常可以使用 sprintf("%u", filesize($file))来克服此问题。

多学两招：

在 PHP 内置的文件系统操作函数中，不仅可以获取到文件的类型、大小和修改时间，而且可以获取到文件的上次访问时间（fileatime()）、inode 修改时间（filectime()）、文件的组（filegroup()）、文件的所有者（fileowner()）和文件的权限（fileperms()）等。

实例 202　将文本文件中的内容存储到数据库中

（实例位置：配套资源\SL\11\202）

实例说明

文本文件中的数据也可以转存到数据库中。例如，在编程词典服务网中，有一个编程词典系列软件注册信息提交页面，在该页面中编程词典用户不但提交个人信息，而且将安装编程词典生成的注册信息文件提交到服务器中，在提交注册信息的同时，将注册信息中的数据与用户个人信息一起存储到数据库中。本实例模拟这个功能，开发一个将文本文件上传到服务器，并且将文本文件中数据转存到数据库中的实例。其运行结果如图 11.13 所示。



图 11.13　将文本文件中的内容存储到数据库中





实现过程

具体步骤如下：

（1）创建 index.php 文件。首先，设计网页页面。然后，创建一个表单，通过文件域提交要上传的 ini 文件。接着，获取表单中提交的文件，将文件存储到服务器指定的文件夹下，通过 file_get_contents()函数读取上传文件的内容，并且将读取的结果存储到指定的数据表中。最后，执行查询语句，输出数据库中存储的 ini 文件的内容。其关键代码如下：

```
<?php
if (! empty ( $_FILES ['up_picture'] ['name'] )) {          //判断上传内容是否为空
  $type = $_FILES ['up_picture'] ['name'];
  $types = strstr ( $type, '.' );                           //获取文件后缀
  if ($types == ".ini") {
       if ($_FILES ['up_picture'] ['error'] > 0) {          //判断文件是否可以上传到服务器
            echo "上传错误:";
            switch ($_FILES ['up_picture'] ['error']) {
                 case 1 :
                      echo "上传文件大小超出配置文件规定值";
                      break;
                 case 2 :
                      echo "上传文件大小超出表单中约定值";
                      break;
                 case 3 :
                      echo "上传文件不全";
                      break;
                 case 4 :
                      echo "没有上传文件";
                      break;
            }
       } else {
            if (! is_dir ( "./upfile/" )) {                 //判断指定目录是否存在
                 mkdir ( "./upfile/" );                     //创建目录
            }
            $file_name = time () . strstr ( $_FILES ['up_picture'] ['name'], '.' );//定义上传文件名称
            $path = './upfile/' . $file_name;               //定义上传文件名称和存储位置
            if (is_uploaded_file ( $_FILES ['up_picture'] ['tmp_name'] )) {  //判断文件是否是 HTTP
POST 上传
                 if (! move_uploaded_file ( $_FILES ['up_picture'] ['tmp_name'], $path )) { //执行
上传操作
                      echo "上传失败";
                 } else {
                      //读取文本文件中数据，并且实现转义和编码的转换
                      $arr = iconv ("gb2312", "utf-8", addslashes ( file_get_contents ( $path ) ) );
                      include_once ("conn/conn.php");         //连接数据库
                      $sql  =  "insert  into  tb_files(file_name,file_content,file_date)values('$file_
```





```
name','$arr','" . date ( "Y-m-d H:i:s" ) . "')";                    //定义 SQL 添加语句
                                $result = mysql_query ( $sql, $conn );          //执行添加操作
                                echo "文件：  " . $_FILES ['up_picture'] ['name'] . "  上传成功!" . "<br>";
                                echo "大小：  " . $_FILES ['up_picture'] ['size'] . "  字节";
                        }
                } else {
                        echo "上传文件" . $_FILES ['up_pictute'] ['name'] . "不合法！ ";
                }
        }
    } else {
        echo "上传文件" . $_FILES ['up_pictute'] ['name'] . "类型不正确！ ";
    }
}
?>
```

（2）创建 conn 文件夹，编写 conn.php 文件，连接数据库服务器，连接 db_database11 数据库。其代码如下：

```
<?php
$conn = mysql_connect ( "localhost", "root", "111" ) or die ( "连接数据库服务器失败！ " . mysql_
error () );
mysql_select_db ( "db_database11", $conn );          //选择数据库 db_database11
mysql_query ( "set names utf8" );                    //设置数据库编码格式 UTF-8
?>
```

技术要点

在本实例中，运用了三个方面的技术：第一将文本文件上传到服务器；第二读取服务器指定文件夹下的文本文件中的数据；第三连接数据库，将从文本文件中读取的数据存储到指定数据表中。

其中，最为关键的是将文本文件中的数据存储到数据库中，在执行数据的读取和存储时，有两个细节必须把握。第一个，应用 addslashes()函数，对从文本文件中读取的数据进行转义，因为在读取的数据中可能包含单引号（'）、双引号（"）、反斜线（\）或者 NUL（NULL 字符），如果不对其进行转换，在执行 SQL 语句时可能会出现错误。

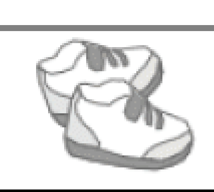
addslashes()函数，通过反斜线来引用字符串。语法如下：

```
string addslashes ( string str)
```

在其返回的字符串中，为了数据库查询语句等的需要在某些特定字符前加上了反斜线。这些特定字符包括：单引号（'）、双引号（"）、反斜线（\）和 NUL（NULL 字符）。

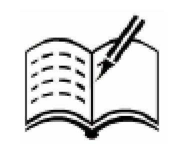
指点迷津：

默认情况下，PHP 指令会自动对所有的 GET、POST 和 COOKIE 数据运行 addslashes()函数。所以，如果是通过 GET 或者 POST 方法提交的数据是不需要进行转义的，可以直接添加到数据库中；如果提交的数据是从文本文件中读取或者自定义的内容，那么就有必要对其进行转义。



多学两招：

将从文本文件中读取的数据存储到数据库时，必须应用 addslashes()函数对文本文件返回的字符串进行转义，否则将会出现错误。



Note

实例 203　判断文件是否被改动

（实例位置：配套资源\SL\11\203）

实例说明

在网站的管理系统中，有时需要查看某个文件是否被修改过，在什么时间被修改的，最后的改变时间是什么时候。本实例就可以实现这个功能，对表单中提交的文件进行判断，检测出它的修改时间，其运行结果如图 11.14 所示。



图 11.14　判断文件是否被改动

实现过程

首先，创建 PHP 文件，并应用图片布局设置网页页面。然后，在创建的 PHP 文件内创建一个表单，通过文件域提交要判断的文件。接着，获取表单中提交的文件路径，应用 filectime()和 filemtime()函数对提交的文件进行检测。最后，当用户单击“浏览”按钮选取本地文件后，继续单击“提交”按钮即可输出检测结果。

index.php 文件的关键代码如下：

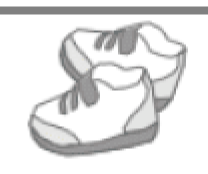
```
<?php
if(isset($_POST['files'])){                                    //判断用户是否提交文件
$file = iconv ( "utf-8", "gb2312", $_POST ['files'] );         //实现编码格式的转换
if (file_exists ( $file )) {                                   //判断文件是否存在
        $change_time = filectime ( $file );                    //获取文件的最后 incode 时间
        $time = date ( "Y-m-d h:i:s", $change_time );          //时间戳的格式化
        $last_time = filemtime ( $file );                      //获取文件的最后修改时间
        $times = date ( "Y-m-d h:i:s", $last_time );           //时间戳的格式化
} else {
        $result = "该文件不存在!!";
}
}
?>
```

技术要点

本实例主要应用 filectime()和 filemtime()函数，检测文件的 incode 最后改变时间和最后的修改时间，并应用 date()函数对检测返回的时间戳进行格式化。

filectime()函数，返回指定文件 filename 的 inode 最后改变时间。语法如下：

```
int filectime(string filename);
```

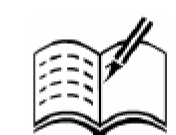


成功则返回 UNIX 时间戳，否则返回 false。

filemtime()函数，返回指定文件 filename 的最后修改时间。语法如下：

```
int filemtime(string filename);
```

成功则返回 UNIX 时间戳，否则返回 false。



多学两招：

通过文件系统函数不但可以获取到文件的最后修改时间，而且可以获取到文件的最后访问时间，其应用的是 fileatime()函数。

实例 204　目录操作汇总

（实例位置：配套资源\SL\11\204　视频位置：配套资源\SP\11\204）

实例说明

在前面的实例中，已经涉及到很多目录操作的方法，例如，创建、获取当前目录、删除目录等。在本实例中，将对目录的基本操作进行一次汇总，让读者对目录操作方法有一个系统的了解。运行本实例，可以实现目录的创建、浏览和删除操作，其运行结果如图 11.15 所示。



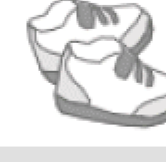
图 11.15　目录操作汇总

实现过程

具体步骤如下：

（1）创建 index.php 文件。通过 switch 语句编写一个网页框架，根据超链接中传递的参数值，在 index.php 页面中完成创建目录、浏览目录、删除目录和删除文件功能之间的跳转。其关键代码如下：

```
<?php
switch ($_GET ['dir']) {                                    //获取 dir 路径
```

```
        case "create" :                                              //根据获取路径加载功能文件
            include ("create.php");
            break;
        case "look" :
            include ("look.php");
            break;
        case "deletedir" :
            include ("deletedir.php");
            break;
        case "deletefile" :
            include ("deletefile.php");
            break;
        default :
            include ("look.php");
            break;
    }
    ?>
```

（2）创建 create.php 文件，应用 mkdir()函数完成目录的创建。其关键代码如下：

```
<?php
if （isset($_POST ['Submit'] )) {                                    //判断用户是否执行提交操作
  $file_dir = iconv ( "utf-8", "gb2312", $_POST ['objfile'] );       //将文件名称编码格式转换
  if (is_dir ( $file_dir )) {                                         //判断创建的目录是否已经存在
        echo "<script>alert('目录已经存在!');</script>";              //提示用户目录已经存在
  } else {
        if (mkdir ( $file_dir )) {                                    //创建文件目录
              echo "<script>alert('目录创建成功!');</script>";        //提示用户目录创建成功
        } else {
              echo "<script>alert('目录创建失败!');</script>";        //提示用户目录创建失败
        }
  }
}
?>
```

（3）创建 look.php 文件，通过 getcwd()函数定位到当前目录，并应用 opendir()函数和 readdir()函数读取当前目录中的内容，其代码请参考本书配套资源。

（4）创建 deletedir.php 文件，在 look.php 文件的基础上，为输出的目录创建删除超链接，链接到 delete.php 文件，完成目录的删除操作。

（5）创建 deletefile.php 文件，在 look.php 文件的基础上，为输出的文件创建删除超链接，链接到 delete.php 文件，完成文件的删除操作。

（6）创建 delete.php 文件，分别应用 rmdir()函数和 unlink()函数完成目录和文件的删除。

技术要点

本实例主要应用 mkdir()函数、is_dir()函数、getcwd()函数、rmdir()函数、opendir()函数和 readdir()函数等，完成目录的判断、创建、打开、读取和删除操作。所使用函数的功能和语法如表 11.1 所示。



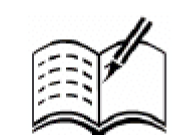


表 11.1　目录操作函数汇总

名　称	语　法	功　能
mkdir()	bool mkdir(string pathname [, int mode])	新建一个由 pathname 指定的目录。参数 mode 指定目录的模式，Mode 在 Windows 下被忽略。自 PHP 4.2.0 起成为可选项。默认的 mode 是 0777，意味着最大可能的访问权
is_dir()	bool is_dir (string filename)	如果文件名存在并且为目录则返回 true。如果 filename 是一个相对路径，则按照当前工作目录检查其相对路径
getcwd()	string getcwd (void)	返回当前的工作目录
rmdir()	bool rmdir (string dirname)	删除 dirname 所指定的目录。该目录必须是空的，而且要有相应的权限。如果成功则返回 true，失败返回 false
opendir()	resource opendir (string path [, resource context])	打开一个目录句柄，成功返回目录句柄的 resource，失败返回 false。返回值可用于 closedir()、readdir()和 rewinddir()函数中。如果参数 path 不是一个合法的目录或者因为权限限制或文件系统错误而不能打开目录，opendir()返回 false 并产生一个 E_WARNING 级别的 PHP 错误信息。可以在 opendir()前面加上“@”符号来抑制错误信息的输出
readdir()	string readdir (resource dir_handle)	从目录句柄中读取条目，成功则返回目录中下一个文件的文件名，否则返回 false。其参数 dir_handle 是目录句柄的 resource，之前由 opendir()打开

多学两招：

在应用 rmdir()函数删除指定的目录时，被删除的路径必须是空的目录，并且权限必须要合乎要求，否则将返回 false。



面向对象编程

本章读者可以学到如下实例：

- ⏭ 实例 205　使用类的属性保存数据库连接参数
- ⏭ 实例 206　数据库连接类中定义数据库连接方法
- ⏭ 实例 207　数据统计类中定义求数值平方的方法
- ⏭ 实例 208　使用$this 关键字调用汽车类自身的方法
- ⏭ 实例 209　学生类中使用构造方法为学生信息初始化
- ⏭ 实例 210　汽车类使用 public 关键字定义汽车的行驶方法
- ⏭ 实例 211　使用 private 关键字定义汽车的颜色属性
- ⏭ 实例 212　使用 protected 关键字定义汽车的保修年限
- ⏭ 实例 213　苹果子类继承水果父类
- ⏭ 实例 214　使用 parent 关键字调用父类的方法
- ⏭ 实例 215　苹果字类中覆盖水果父类中的方法
- ⏭ 实例 216　美食抽象类
- ⏭ 实例 217　学生类多重接口的实现
- ⏭ 实例 218　通过继承实现多态
- ⏭ 实例 219　通过接口实现多态
- ⏭ 实例 220　使用 final 关键字防止类被继承
- ⏭ 实例 221　使用 static 关键字定义类的静态成员
- ⏭ 实例 222　使用 clone 关键字实现对象的克隆
- ⏭ 实例 223　使用__set()方法为类中未经定义的属性赋值
- ⏭ 实例 224　使用__get()方法获取未声明属性的名称
- ⏭ 实例 225　使用__call()方法打印类中未定义方法的信息
- ⏭ 实例 226　使用__toString()方法将类的实例转化为字符串
- ⏭ 实例 227　使用__isset()方法提示未定义属性信息
- ⏭ 实例 228　使用单例模式制作数据库管理类
- ⏭ 实例 229　使用策略模式打印客户端浏览器类型
- ⏭ 实例 230　使用工厂模式设置用户访问权限



实例 205　使用类的属性保存数据库连接参数

（实例位置：配套资源\SL\12\205　视频位置：配套资源\SP\12\205）

实例说明

运行本实例，如图 12.1 所示。首先在图中的文本框内输入连接 MySQL 数据库服务器所需的必选参数，然后单击“连接”按钮即可建立与指定的 MySQL 数据库服务器的连接，并将连接结果打印在页面中。

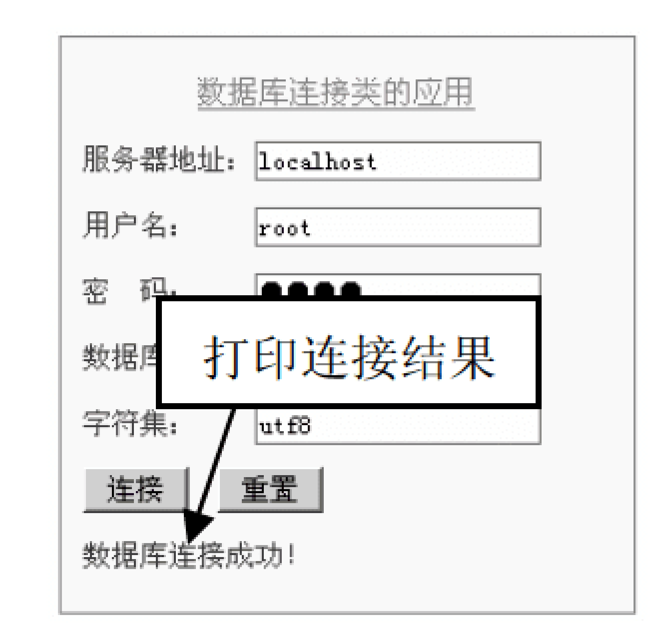


图 12.1　数据库连接成功

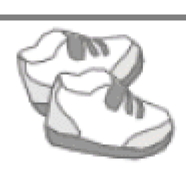
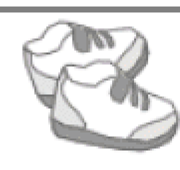
实现过程

实现本实例，首先需要定义数据库连接类，然后建立数据库参数录入表单来指定数据库连接时所需要的连接参数，下面具体讲解该数据库连接类的设计和实现过程。

（1）定义数据库连接类，并通过类的属性保存连接 MySQL 数据库所需的参数。该类的具体实现代码如下：

```
class ConnDb
{
    private $host;                          //MySQL 数据库服务器地址
    private $username;                      //用户名
    private $password;                      //密码
    private $charset;                       //数据库编码
    private $dbname;                        //数据库的名称
    public function __construct ($host, $username, $password, $dbname, $charset = 'utf-8')      //构造
方式实现类的初始化
    {
        $this->host = $host;                //初始服务器地址
        $this->username = $username;        //初始用户名
     $this->password = $password;           //初始用户密码
     $this->dbname = $dbname;
        $this->charset = $charset;          //初始数据库字符集
    }
    public function getConnId ()            //数据库连接方法
    {
        @$connId = mysql_connect($this->host, $this->username, $this->password);        //获得数
据库连接句柄
        @mysql_select_db($this->dbname, $connId);
        @mysql_query('set names ' . $this->charset);    //设置字符集
        return $connId;                     //返回连接句柄
    }
}
```

上述代码中，定义了$host、$username、$password、$charset 和$dbname 5 个私有属性，

分别用来保存数据库服务器的主机地址或名称、数据库服务器用户名、用户密码、字符集和要连接的数据库名，并通过构造方法__construct()实现数据库连接参数的初始化，最后定义 getConnId()方法返回数据库连接句柄。

（2）建立数据库连接参数录入表单。该过程通过 HTML 语言的表单标签实现，具体实现代码详见本书附带配套资源。

（3）当单击数据库参数录入表单中的“连接”按钮，将会使用如下代码判断是否成功连接上数据库。

```
if(isset($_POST['host'])){
    require 'ConnDb.php';                                    //包含数据库连接类文件
    $connDb = new ConnDb($_POST['host'], $_POST['username'], $_POST['password'],
$_POST['dbname'], $_POST['charset']);  //实例数据库连接类
    echo !$connDb->getConnId()?'数据库连接失败！':'数据库连接成功！';      //打印连接结果
}
```

上述代码中，首先通过 isset($_POST['host'])判断用户是否已经提交了表单，如果是则使用 require()语句包含数据库连接类，然后使用 new 关键字实例该类，并通过调用类中的 getConnId()方法来返回数据库连接状态，最后打印出连接结果。

多学两招：

用“@”屏蔽运行期错误。

在程序运行时，有时会难以避免地出现运行时错误并在页面中打印错误信息，这样不仅会使页面的友好性大打折扣，而且还会对程序的安全带来极大隐患。使用 try/catch 语句可以屏蔽运行期错误，不过需要书写的代码量相对较大。PHP 提供了“@”关键字来屏蔽运行期错误，只要在可能发生运行期错误代码前加上“@”即可，例如，本实例用来屏蔽连接数据库失败时的错误提示代码如下：

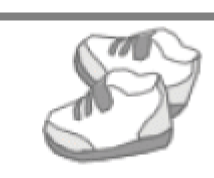
```
@$connId = mysql_connect($this->host, $this->username, $this->password);      //获得数据库连接句柄
```

技术要点

类是对事物的抽象，在对类进行定义时合理地设置类的属性和方法较为重要。例如，在本实例所应用的数据库连接类中，将数据库服务器地址、数据库服务器用户名、密码、数据库名称以及数据库字符集作为类的属性保存，而将数据库连接的实现过程定义为类的方法，这是因为数据库连接参数在类的生命周期内基本没有变化，而且属于特性范围，所以定义为类的属性，而数据库连接方法属于类的动作或功能，所以定义成类的方法。

在 PHP5.0 以前版本中，类属性使用关键字 dim 声明，通过该关键字声明的类属性可以在类声明周期的任何范围内被调用。而在 PHP5.0 以后版本中引入了 public、protected 和 private 关键字来对类中属性或方法进行声明，这样根据不同的关键字将类的属性或方法划分为公有成员、保护成员和私有成员 3 种，其中不同关键字修饰的类成员的说明和区别如下：

☑ public：表示该成员可以在任何范围被调用，包括类体内部、该类的子类和类实



例的对象。

☑ protected：表示该方法可以在类中或该类的子类中被调用。

☑ private：表示该类只能在类体内被调用。

指点迷津：

如果没有对类中方法指定任何访问权限，则 PHP 会默认该方法的访问权限为 public。

实例 206　数据库连接类中定义数据库连接方法

（实例位置：配套资源\SL\12\206　视频位置：配套资源\SP\12\206）

实例说明

当使用 PHP 的 MySQL 函数库管理 MySQL 数据库时，首先需要获得数据库连接句柄，然后才能进一步进行增、删、改、查操作。运行本实例，首先在图 12.2 所示的表单中输入数据库连接参数，然后单击“连接”按钮，如果成功连接上 MySQL 数据库则会在页面中打印出数据库连接句柄。

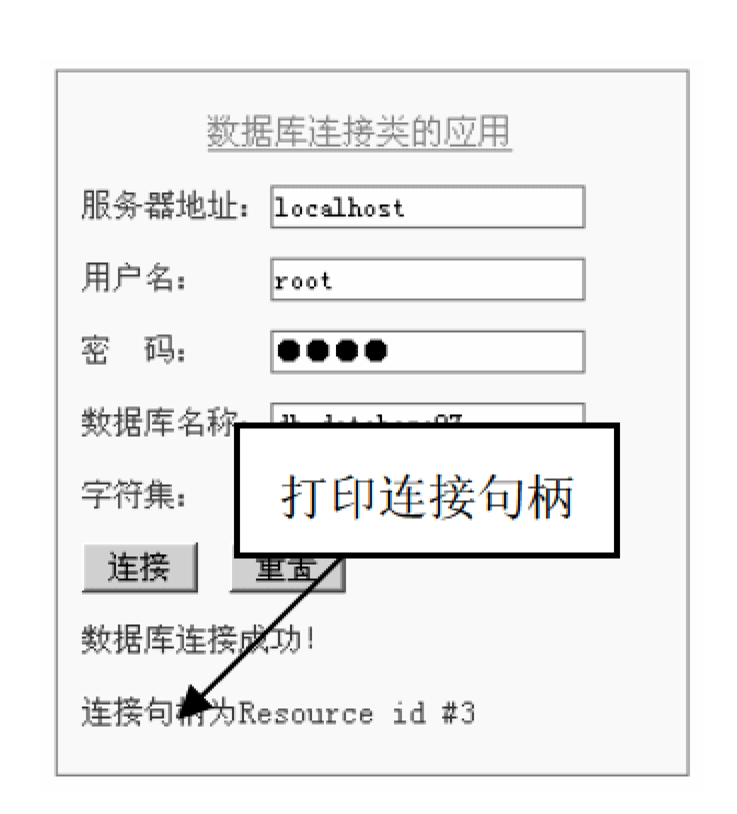


图 12.2　打印连接句柄

实现过程

具体步骤如下：

（1）定义数据库连接类，在类中定义数据库连接的方法 getConnId()，实现该过程的代码如下：

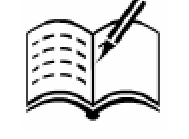
```
public function getConnId ()                                    //数据库连接方法
{
    @$connId = mysql_connect($this->host, $this->username, $this->password); //获得数据库连接句柄
    @mysql_select_db($this->dbname, $connId);
    @mysql_query('set names ' . $this->charset);                //设置字符集
    return $connId;                                             //返回连接句柄
}
```

在上述方法中，首先通过函数 mysql_connect()获得数据库连接句柄，然后使用函数 mysql_select_db()选择要连接的数据库，并使用函数 mysql_query()执行查询语句来设置数据库的字符集。

（2）实例数据库连接类，并打印连接句柄，实现代码如下所示：

```
$connDb = new ConnDb($_POST['host'], $_POST['username'], $_POST['password'], $_POST['dbname'], $_POST['charset']);    //实例数据库连接类
echo !$connDb->getConnId()?'数据库连接失败！':'数据库连接成功！<br/>连接句柄为'.$connDb->getConnId();    //打印连接句柄
```







多学两招：

通过“set names utf8”语句动态设置数据库的字符集为UTF-8编码。

在对数据库内容进行检索输出时，有时可能会出现乱码现象，一般都是数据库字符集设置存在偏差所致的，为了避免上述情况的发生，可以在执行数据库查询语句前执行“set names 编码”语句，在开发项目时，为了便于国际化，建议应用UTF-8编码。

脚下留神：

设置MySQL数据库为UTF-8编码所执行的语句应该为“set names utf8”，而非“set names utf-8”，即“utf”和“8”之间没有“-”。

技术要点

在PHP面向对象的编程方式中，当函数被定义在类中，就被称为成员方法，成员方法的功能应该匹配该类的特定功能。

类中定义的成员方法不能出现重名的情况，并且需要注意以下几点：

（1）函数名称不区分大小写，例如myfun()与Myfun()指的是同一个函数。

（2）函数的参数没有限制，可以定义任意个需要的参数

（3）函数名称理论上可以使用汉字，但汉字是双字节字符，为了避免出现无法预知的问题，尽量不使用。

实例207　数据统计类中定义求数值平方的方法

（实例位置：配套资源\SL\12\207）

实例说明

运行本实例，如图12.3所示。首先在图的文本框中输入一组数字并用半角逗号分割，然后单击“计算”按钮将在页面中打印出该组数字的平均值。

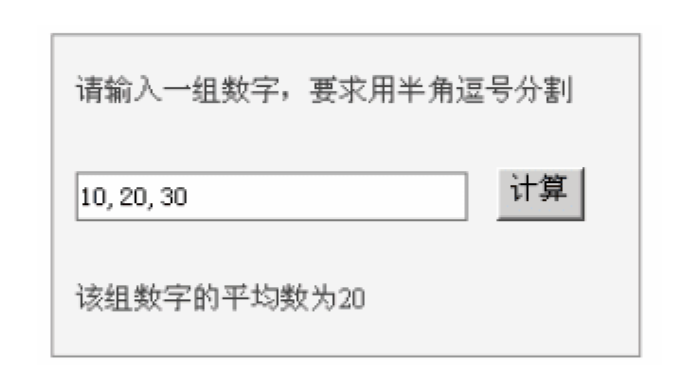


图12.3　计算一组数的平均值

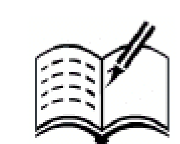
实现过程

（1）建立数据统计类Stat，并在类体内定义静态方法getAvg()用来实现求多个数字的平均值，其实现代码如下：

```
class Stat
{
    public static function getAvg ($arryNum)                //定义静态方法来求平均值
    {
        $totalNum = count($arryNum);                        //获得数字总数
        if ($totalNum == 0) {                               //如果数字个数为 0 则返回 null
            return null;
        } else {
            $sum = 0;                                       //用变量$sum 保存所有数字的和
```



```
                for ($i = 0; $i < $totalNum; $i ++) {          //通过循环计算所有数字的和
                    $sum += $arryNum[$i];
                }
                return $sum / $totalNum;                      //返回平均数
            }
        }
    }
```



上述代码用于实现求平均数的 getAvg()方法中，为了能够实现求任意个数字的平均数，将数组作为参数，这样只要求得该数组所有元素的平均值即可，在具体实现过程中，首先使用函数 count()计算数组元素的个数，然后使用 for 循环语句计算数组中所有元素的和，最后使用 return 语句返回所求得的平均数。

（2）建立数字录入表单，当用户单击表单中的“计算”按钮时，将通过如下代码计算在表单中所录入数字的平均数。

```
    if(isset($_POST['nums']) && trim($_POST['nums'])!=''){          //判断是否已经提交表单
        require 'Stat.php';                                          //包含数据统计类
        $arrayNum = explode(',', $_POST['nums']);                     //将提交数字保存到数组中
        echo '该组数字的平均数为' . Stat::getAvg($arrayNum);          //调用统计类的静态方法 getAvg()
求平均数
    }
```

上述代码中，首先判断用户是否提交了表单，如果是则首先使用 require 语句包含数据统计类，然后使用函数 explode()用半角逗号分割所提交数字字符串，最后调用数据统计类的静态方法 getAvg()计算所录入数字的平均数。

指点迷津：

函数 explode()用于实现将字符串以指定的字符或子串分割，并将分割结果保存到数组中。

多学两招：

为了提高代码运行效率，建议将 count()函数放在循环体外。

本实例计算数组中所有元素的和可以有两种写法，分别如下：

将函数 count()写在 for 循环体内：

```
for($i=0; $i<count($arrayNum); $i++){}
```

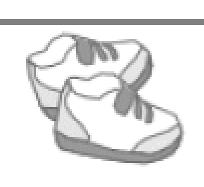
将函数 count()写在 for 循环体外：

```
$count = count($arrayNum);
for($i=0; $i<$count; $i++){}
```

查看上述两种写法可知，第一种写法在执行每次循环时都需要执行一次函数 count()，而第二种写法只在执行循环前执行一次函数 count()，所以在程序开发过程中，建议采用第二种写法，可以提高程序运行效率。

技术要点

实现本实例时，由于该类只包含一个用于计算平均数的方法，所以在实现该类时将求



平均数的方法定义为静态方法，这样在该类的情况下通过类名即可直接调用该方法。

在 PHP 面向对象编程方式中，静态方法隶属于某个类，但不受类的束缚，可以直接被外部访问。与定义类中普通方法相比，定义静态方法时需要使用 static 关键字进行修饰，其定义格式如下所示：



Note

```
public static function 方法名()
{
    //该方法实现的功能
}
```

在类体内部调用该方法时，使用 self 关键字加“::”调用，在类外直接使用类名加“::”引用。

实例 208　使用$this 关键字调用汽车类自身的方法

（实例位置：配套资源\SL\12\208　视频位置：配套资源\SP\12\208）

实例说明

本实例通过$this 关键字在汽车类内部调用设置汽车颜色的方法和设置汽车品牌的方法来输出汽车的信息。运行本实例，如图 12.4 所示，分别选择表单中的颜色和类型单选按钮，然后单击“提交”按钮即可在图中打印出汽车信息。

请选择颜色：
红色： 白色： 黑色： 宝石蓝：
请选择类型：
奔驰： 宝马： 奥迪： 捷达：
提交
我的汽车是黑色宝马

图 12.4　打印汽车信息

实现过程

具体步骤如下：

（1）定义汽车类 Car。首先在该类中定义汽车颜色标识和汽车类别标识两个属性，然后使用构造函数对这两个属性进行初始化，并在类内定义 getColor()方法和 getType()方法，分别用来获取汽车颜色和汽车类型，最后使用 getInfo()方法打印出汽车信息。本实例的汽车类实现代码如下：

```
class Car                                                   //定义汽车类
{
    private $colorFlag;                                     //颜色标识
    private $typeFlag;                                      //类型标识
    public function __construct ($colorFlag, $typeFlag)     //构造方法
    {
        $this->colorFlag = $colorFlag;                      //颜色标识初始化
        $this->typeFlag = $typeFlag;                        //类型标识初始化
    }
    public function getColor ()                             //定义获取颜色方法
    {
        switch ($this->colorFlag) {           //使用 switch 语句根据不同颜色获得标识颜色
            case 0:
```



```
                    $color = '红色';
                    break;
                case 1:
                    $color = '白色';
                    break;
                case 2:
                    $color = '黑色';
                    break;
                default:
                    $color = '宝石蓝';
            }
            return $color;
        }
        public function getType ()                          //定义获得汽车类型方法
        {
          switch ($this->typeFlag) {                        //根据类型标识获得汽车类型
                case 0:
                    $type = '奔驰';
                    break;
                case 1:
                    $type = '宝马';
                    break;
                case 2:
                    $type = '奥迪';
                    break;
                default:
                    $type = '捷达';
            }
            return $type;
        }
        public function getInfo ()                              //获得汽车信息
        {
            return '我的汽车是' . $this->getColor() . $this->getType();   //调用类内方法返回汽车信息
        }
    }
```

（2）建立汽车属性选择表单，当用户单击表单的“提交”按钮后将通过如下代码接收表单提交的颜色标识和类别标识，然后使用 new 关键字对汽车类进行实例化，最后使用实例化的汽车类的对象调用该类的 getInfo()方法打印汽车信息。

```
    if(isset($_POST['color']) && isset($_POST['type'])){        //判断是否提交了表单
        require 'Car.php';                                      //包含汽车类
        $colorFlag = $_POST['color'];                           //获得表单提交的颜色标识
        $typFlag = $_POST['type'];                              //获得表单提交的类别标识
        $car = new Car($colorFlag, $typFlag);                   //对汽车类进行实现化
        echo $car->getInfo();                                   //打印汽车信息
    }
```

技术要点

在 PHP 面向对象编程方式中，可以使用$this 关键字对类自身的属性或方法进行调用。





例如，已经在A类中定义了方法fun1()，在该类的fun2()方法中可以使用如下方式对fun1()进行调用。

```
class A
{
    private function func1()
    {
                    //fun1()实现的功能
    }
    public function fun2()
    {
        $this->fun1();      //调用fun1()
    }
}
```

实例209　学生类中使用构造方法为学生信息初始化

（实例位置：配套资源\SL\12\209）

实例说明

本实例采用二维数组模拟数据库存储学生信息，然后通过for循环语句实例学生类，并通过构造方法为学生类的属性赋初值，最后通过类的getXxx()方法获得学生属性，并将学生信息输出在页面，如图12.5所示。

学号	姓名	年龄	住址
0312310	小明	16	北京西城区
0312311	小张	16	北京宣武区
0312312	小赵	17	北京海淀区

图12.5　学生信息列表

实现过程

具体步骤如下：

（1）首先建立二维数组$students用来模拟数据库，代码如下：

```
$students = array(                              //二维数组，模拟数据库
    0 => array('0312310', '小明', '16', '北京西城区') ,
    1 => array('0312311', '小张', '16', '北京宣武区') ,
    2 => array('0312312', '小赵', '17', '北京海淀区')
);
```

（2）定义学生类Student，在该类中使用构造方法__construct()实现对学生信息的初始化，然后针对类的各个属性建立相应的getXxx()方法来获取属性的值，代码如下：

```
class Student
{
    private $id;                                //学生ID
    private $name;                              //学生名称
    private $age;                               //学生年龄
```

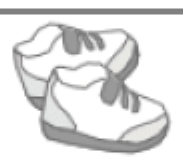




```
        private $address;                                     //学生住址
        public function __construct ($id, $name, $age, $address)  //构造方法，对学生信息初始化
        {
            $this->id = $id;
            $this->name = $name;
            $this->age = $age;
            $this->address = $address;
        }
        public function getId ()                              //获得学生 ID
        {
          return $this->id;
        }
        public function getName ()                            //获得学生名称
        {
            return $this->name;
        }
        public function getAge ()                             //获得学生年龄
        {
            return $this->age;
        }
        public function getAddress ()                         //获得学生住址
        {
            return $this->address;
        }
    }
```

（3）通过 for 循环语句遍历保存学生信息的$students 数组，遍历过程中将学生信息保存到 Student 学生类中，并通过该类实例的对象调用类中相应的 getXxx()方法，获取学生信息。实现该过程的代码如下：

```
<?php
require 'arrayDb.php';
require 'Student.php';
$count = count($students);
?>
<table border="0" align="center" cellpadding="0" cellspacing="1" bgcolor="#009933">
  <tr>
    <td width="100" height="22" bgcolor="#009933"><font color="#FFFFFF">学号</font></td>
    <td width="100" bgcolor="#009933"><font color="#FFFFFF">姓名</font></td>
    <td width="100" bgcolor="#009933"><font color="#FFFFFF">年龄</font></td>
    <td width="200" bgcolor="#009933"><font color="#FFFFFF">住址</font></td>
  </tr>
<?php
for ($i=0; $i<$count; $i++){
    $stu = $students[$i];
    $student = new Student($stu[0], $stu[1], $stu[2], $stu[3]);
?>
  <tr>
```





Note

```
            <td height="22" bgcolor="#FFFFFF"><?= $student->getId()?></td>
            <td bgcolor="#FFFFFF"><?= $student->getName()?></td>
            <td bgcolor="#FFFFFF"><?= $student->getAge()?></td>
            <td bgcolor="#FFFFFF"><?= $student->getAddress()?></td>
        </tr>
    <?php
    }
    ?>
```

多学两招：

PHP5.0 中构造方法名称以双下划线开头。

在使用 PHP5.0 的构造方法时，一定要注意构造方法__construct()是以双下划线开头，而非单下划线。

构造方法是类中的一个特殊函数，当创建一个类的实例时，将会自动调用类的构造方法。在 PHP5.0 以前的版本中构造方法名称与类名相同，在 PHP5.0 以后的版本中统一使用__construct()作为构造方法名称，使用构造方法应该注意如下两点：

☑ 构造方法没有返回值类型和返回值，这是因为构造方法是在创建对象时自动调用的，并不是一个独立的函数，因此不需要返回值。

☑ 构造方法的主要功能是实现对类的初始化工作。

实例 210　汽车类使用 public 关键字定义汽车的行驶方法

（实例位置：配套资源\SL\12\210）

实例说明

本实例用于展示在类体外、类内部的方法中和类的子类中调用被 public 关键字所修饰的方法的结果，如图 12.6 所示。

（1）通过汽车类对象调用汽车类的行驶方法的结果：
行驶
（2）通过汽车类的getStatus()方法调用汽车行驶方法的结果：
汽车目前正在行驶
（3）通过汽车类的子类调用汽车类的汽车行驶方法的结果：
小汽车行驶

图 12.6　被 public 关键字修饰的方法在不同范围内被调用

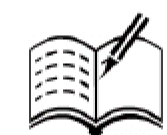
实现过程

具体步骤如下：

（1）首先建立汽车类 Car，并在该类内部定义汽车行驶方法 run()以及获得汽车状态的方法 getStatus()，然后定义类 SmallCar，使该类继承自类 Car，并在该类中定义 smallCarRun()方法。实现该过程的代码如下：

```
    class Car                                   //定义汽车类 Car
    {
        public function run ()                  //定义行驶方法
        {
            return '行驶';
        }
```






```
        public function getStatus ()                //定义返回汽车状态方法
        {
            return '汽车目前正在' . $this->run();
        }
    }
    //定义小汽车类 SmallCar，并使其继承汽车类 Car
    class SmallCar extends Car
    {
        public function smallCarRun ()              //定义小汽车行驶方法
        {
            return '小汽车' . $this->run();
        }
    }
```

（2）打印在不同范围内调用汽车类的行驶方法的结果，代码如下：

```
    require 'Car.php';
    $car = new Car();
    echo '（1）通过汽车类对象调用汽车类的行驶方法的结果：<br/>';
    echo $car->run() . '<br/>';
    echo '（2）通过汽车类的 getStatus()方法调用汽车行驶方法的结果：<br/>';
    echo $car->getStatus() . '<br/>';
    echo '（3）通过汽车类的子类调用汽车类的汽车行驶方法的结果：<br/>';
    $smallCar = new SmallCar();
    echo $smallCar->smallCarRun() . '<br/>';
```

技术要点

在 PHP5.0 面向对象的编程中，被关键字 public 修饰的方法被称为公有方法。公有方法可以被类实例的对象在类内部以及类的子类中被调用。

实例 211　使用 private 关键字定义汽车的颜色属性

（实例位置：配套资源\SL\12\211）

实例说明

本实例主要通过在不同范围内调用类中私有属性来演示私有成员的作用范围。运行本实例，如果通过类实例后的属性直接调用类的私有属性，将在页面中打印如图 12.7 所示的错误信息。如果通过类中方法调用类的私有属性将输出汽车的颜色，如图 12.8 所示。

通过类实例的对象调用类的私有属性的结果：

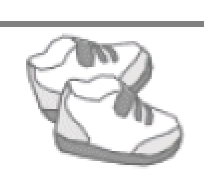
Fatal error: Cannot access private property Car::$color in **D:\AppServ\www\MR\07\011\index.php** on line **17**

图 12.7　通过对象调用类中私有属性

在类中的方法中调用类的私有属性的结果：

汽车颜色是：红色

图 12.8　通过类中方法调用类的私有属性





实现过程

具体步骤如下：

（1）首先定义汽车类 Car，在汽车类中定义私有属性$color，并通过构造方法为$color属性进行初始化，最后定义 getColor()方法返回汽车颜色的值。该过程实现代码如下：

```
class Car                                          //定义汽车类 Car
{
    private $color;                                //汽车颜色
    public function __construct ($color)           //构造方法对类中属性初始化
    {
      $this->color = $color;                       //汽车颜色初始化
    }
    public function getColor ()                    //定义获得汽车颜色的方法
    {
        return $this->color;                       //返回汽车颜色
    }
}
```

（2）对汽车类进行实例化，并指定汽车颜色为红色，然后通过实例的对象调用汽车类中的 getColor()方法获得汽车颜色，该过程实现代码如下所示。在下述代码中，被注释的代码用于实现通过类实例的对象调用类中私有成员的结果，如果去掉注释将出现如图 12.7 所示的错误提示，从而说明类中的私有方法不能被类实例的对象所调用。

```
require 'Car.php';                                         //包含汽车类
$color = '红色';                                           //指定汽车颜色
$car = new Car($color);                                    //对汽车类实例化
//echo '通过类实例的对象调用类的私有属性的结果：<br/>';
//echo @'汽车颜色是：'.$car->color.'<br/>' or die('不能通过类的对象调用类中的私有方法<br/>');
echo '在类中的方法中调用类的私有属性的结果：<br/>';
echo '汽车颜色是：'.$car->getColor().'<br/>';              //打印汽车颜色
```

技术要点

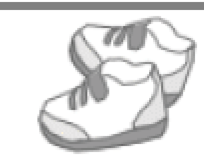
在 PHP5.0 面向对象的编程中，被关键字 private 修饰的成员被称为私有成员，私有成员可以在类体内被调用，但不可以被类实例的对象和类的子类所调用。

实例 212　使用 protected 关键字定义汽车的保修年限

（实例位置：配套资源\SL\12\212）

实例说明

本实例主要用于说明类中的保护成员在不同范围内被调用的结果。运行本实例，当使用类实例的对象调用类中的保护属性时，将出现如图 12.9 所示的错误提示。然后注释掉通过类对象调用类体内保护属性的代码，并分别通过在类体内和类的子类中调用类的保护属性，再次运行该实例将出现如图 12.10 所示的页面。





通过类实例的对象直接调用类中的保护成员的结果

Fatal error: Cannot access protected property Car::$repairTime in **D:\AppServ\www\MR\07\012\index.php** on line **17**

图 12.9　通过对象调用类中保护成员

在类体内调用保护成员的结果

汽车保修年限为：3年

在类子类中调用保护成员的结果

小汽车保修年限为：5年



图 12.10　通过子类调用父类的保护成员

实现过程

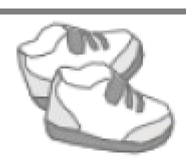
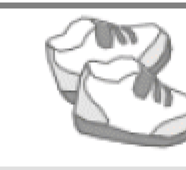
具体步骤如下：

（1）定义汽车类 Car 以及该类的子类 SmallCar。在汽车类中定义汽车的保修时间属性，然后使用构造方法对该属性进行初始化，同时定义获得汽车保修时间的 getRepairTime()方法。在汽车类的子类 SmallCar 中使用构造方法调用其父类的构造方法，从而实现对父类的保修时间属性进行初始化，最后在 SmallCar 类中定义 getSmallCarRepairTime()方法返回小汽车的保修时间。实现该过程的代码如下：

```
class Car                                            //定义汽车类 Car
{
    protected $repairTime;                           //汽车保修时间属性
    public function __construct ($repairTime)        //构造方法对汽车的保修时间进行初始化
    {
        $this->repairTime = $repairTime;
    }
    public function getRepairTime ()                 //获得汽车保修时间的方法
    {
        return $this->repairTime;
    }
}
class SmallCar extends Car                           //定义汽车类的子类 SmallCar
{
    public function __construct ($repairTime)
    {
        parent::__construct($repairTime);            //调用父类构造方法对保修时间属性进行初始化
    }
    public function getSmallCarRepairTime () //调用父类的保修时间属性
    {
        return $this->repairTime;
    }
}
```

（2）使用 new 关键字对汽车类 Car 进行实例化，然后分别打印出在不同范围内调用汽车类保护属性$repairTime 的结果。代码如下所示：

```
require 'Car.php';                                   //包含汽车类
$repairTime = 3;                                     //定义保修年限
$car = new Car($repairTime);                         //对汽车类实例化
//echo '通过类实例的对象直接调用类中的保护成员的结果<br/>';
```

```
//echo '汽车保修年限为：'.$car->repairTime.'年<br/>';
echo '在类体内调用保护成员的结果<br/>';
echo '汽车保修年限为：'.$car->getRepairTime().'年<br/>';
echo '在类子类中调用保护成员的结果<br/>';
$repairTime = 5;                                    //定义保修年限
$smallCar = new SmallCar($repairTime);
echo '小汽车保修年限为：'.$smallCar->getSmallCarRepairTime().'年<br/>';
```

技术要点

在 PHP5.0 面向对象的编程中，被关键字 protected 修饰的成员被称为保护成员，保护成员可以在类体内以及类的子类被调用，但不可以在类体外被调用。

实例 213　苹果子类继承水果父类

（实例位置：配套资源\SL\12\213）

实例说明

本实例首先定义父类水果类，并在该类中定义获得水果颜色和水果形状的方法，然后定义水果类的子类苹果类，在苹果类中只定义构造方法用于实现对类进行初始化，然后通过调用父类的方法获得苹果的颜色和形状。

运行本实例，如图 12.11 所示，首先在表单中输入苹果的颜色和形状属性，然后单击“提交”按钮即可在页面中打印出苹果的颜色和形状属性。

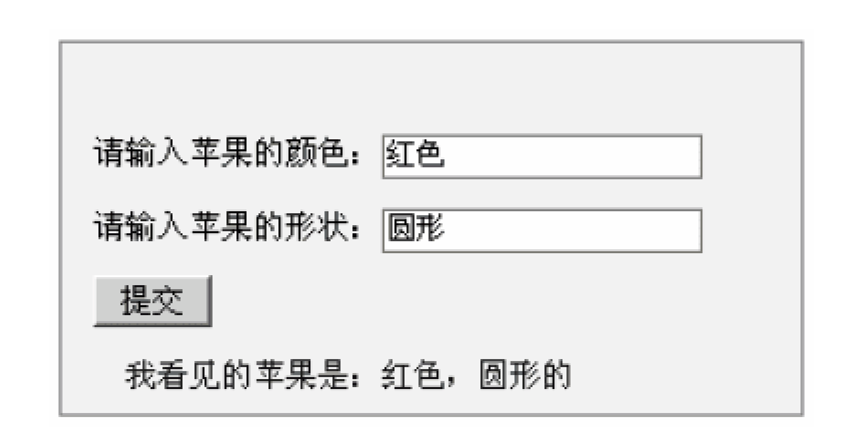


图 12.11　打印苹果的颜色和形状属性

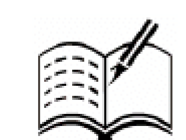
实现过程

具体步骤如下：

（1）定义水果类 Fruit，在该类中定义颜色属性$color 和形状属性$shape，并通过构造方法对这两个属性进行初始化，然后定义用于获得这两个属性的 getColor()方法和 getShape()方法。代码如下：

```
class Fruit                                         //定义水果类
{
    private $color;                                 //颜色属性
    private $shape;                                 //形状属性
    public function __construct ($color, $shape)    //构造方法对水果类初始化
    {
        $this->color = $color;
        $this->shape = $shape;
    }
    public function getColor ()                     //获得水果颜色
```






```
        {
            return $this->color;
        }
        public function getShape ()                      //获得水果形状
        {
            return $this->shape;
        }
    }
```

（2）定义苹果类 Apple，使该类继承自水果类 Fruit，并定义苹果类的构造方法，在构造方法中使用 parent 关键字调用父类构造方法对父类进行初始化。代码如下：

```
    class Apple extends Fruit                            //苹果类继承水果类
    {
        public function __construct ($color, $shape)     //构造方法对类进行初始化
        {
            parent::__construct($color, $shape);
        }
    }
```

（3）建立苹果颜色和形状录入表单。用户在表单中输入苹果的颜色和形状后单击“提交”按钮，将通过如下代码打印苹果的颜色和形状属性。

```
    if(isset($_POST['color']) && $_POST['color']!=''){          //判断是否已经提交表单
        require 'Fruit.php';                                    //包含水果类
        $apple = new Apple($_POST['color'], $_POST['shape']);   //对水果类进行实例化
        echo '     <font color="blue">我看见的苹果是：'.$apple->getColor().'，'.$apple->getShape().'的
</font>';                                                       //打印苹果属性
    }
```

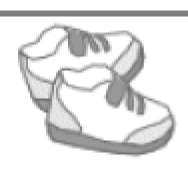
上述代码首先判断用户是否已经提交表单，如果是则首先使用 require 语句包含苹果类，然后使用 new 关键字对苹果类进行实例化，最后使用苹果类实例的对象调用父类中的 getColor()方法和 getShape()方法打印苹果的属性。

技术要点

在 PHP5.0 面向对象编程方式中，可以使用 extends 关键字继承指定的父类。代码如下：

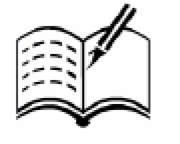
```
    class Parent1                          //定义父类 Parent1
    {
        //父类的成员
    }
    class Child extends Parent1            //定义子类 Child
    {
        //子类的成员
    }
```

通过上述继承关系，即可在子类中使用 parent 关键字加“::”操作符调用父类的公有方法或属性，或者重写非 final 关键字修饰的方法。



实例214　使用parent关键字调用父类的方法

（实例位置：配套资源\SL\12\214）



Note

实例说明

本实例主要使用 parent 关键字在小汽车类中调用汽车类的行驶方法。运行本实例，如图12.12所示，分别在页面中打印出通过汽车类调用其自身的行驶方法和其子类小汽车类调用其自身的行驶方法的结果。

行驶

小汽车可以行驶

图12.12　打印汽车和小汽车的行驶方法

实现过程

具体步骤如下：

（1）定义汽车类，并在该类中定义汽车的行驶方法，代码如下：

```
class Car                              //定义汽车类
{
    public function run ()             //定义行驶方法
    {
        return '行驶';
    }
}
```

（2）定义小汽车类使其继承自汽车类，在类体内定义 smallCarRun()方法，并在该方法中使用 parent 关键字调用汽车类的 run()方法，代码如下：

```
class SmallCar extends Car                    //定义小汽车类，使之继承自汽车类
{
    public function smallCarRun ()            //定义小汽车行驶方法
    {
        return '小汽车可以' . parent::run();
    }
}
```

（3）使用 require 语句包含 Car.php 文件，然后分别对汽车类 Car 和小汽车类 SmallCar 进行实例化，然后分别调用这两个类的行驶方法，代码如下：

```
require 'Car.php';                        //包含汽车类
$car = new Car();                         //对汽车类进行实例化
echo $car->run() . '<br/>';               //调用汽车类行驶方法
$smallCar = new SmallCar();               //实例小汽车类
echo $smallCar->smallCarRun();            //调用小汽车类的行驶方法
```

技术要点

在 PHP5.0 面向对象的编程方式中使用 parent 关键字加操作符“::”调用父类的方法，



调用过程如下所示：

```
class Parent1                                   //定义父类
{
    public function funParent(){                //定义父类的 funParent()方法

}
}
class Child extends Parent1                     //定义子类
{
    public function funChild(){                 //定义子类的 funChild()方法
        parent::funParent();                    //调用父类的方法
}
}
```

使用 parent 关键字不仅可以调用父类的普通方法，而且可以通过调用父类的构造方法实现对父类的初始化。

实例 215　苹果子类中覆盖水果父类中的方法

（实例位置：配套资源\SL\12\215）

实例说明

本实例主要通过在苹果类中重写其父类水果类的 getColor()方法来讲解如何应用类的重写机制。运行本实例，如图 12.13 所示，其中第一行为通过水果类调用其自身的 getColor()方法的输出结果，第二行为苹果类调用其自身的 getColor()方法的输出结果，并且该方法重写了其父类水果类的 getColor()方法。

```
不同的水果颜色不同，无法确定
苹果的颜色是红色
```

图 12.13　重写水果类中的 getColor()方法的结果

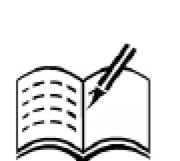
实现过程

具体步骤如下：

（1）定义水果类 Fruit，并在该类中定义 getColor()方法，然后定义水果类的子类 Apple，使该类继承自水果类 Fruit，在苹果类 Apple 中重写其父类的 getColor()方法。实现该过程的代码如下：

```
class Fruit                                     //定义水果类
{
    public function getColor ()                 //获得水果颜色
    {
        return '不同的水果颜色不同，无法确定';
    }
}
class Apple extends Fruit                       //苹果类继承水果类
```






```
{
    private $color;                                 //定义苹果颜色属性
    public function __construct ($color)            //构造方法对类进行初始化
    {
        $this->color = $color;
    }
    public function getColor ()                     //获得水果颜色
    {
        return '苹果的颜色是' . $this->color;
    }
}
```

（2）包含 Fruit.php 文件，然后使用 new 关键字对 Fruit 类进行实例化，并通过实例化的对象调用该类的 getColor()方法，最后对 Apple 类进行实例化，并调用 Apple 类中的 getColor()方法打印苹果的颜色。该过程的实现代码如下所示：

```
require 'Fruit.php';                                //包含 Fruit.php 文件
$fruit = new Fruit();                               //对水果类进行实例化
echo $fruit->getColor().'<br/>';                    //打印水果类的获得颜色方法
$apple = new Apple('红色');                          //对苹果类进行实例化
echo $apple->getColor();                            //打印苹果类的获得颜色方法
```

技术要点

子类不仅可以调用父类的方法，而且可以对指定的方法进行重写，应用重写机制可以扩展类中方法的功能，其中实现对类中方法重写的代码如下：

```
class Parent1                           //父类 Parent1
{
    public function fun(){
                                        //父类的 fun()方法
    }
}
class Child extends Parent1             //子类 Child 继承自 Parent1
{
    public function fun(){
                                        //重写父类的 fun()方法
    }
}
```

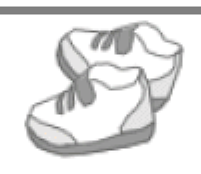
对上述代码中的子类 Child 进行实例化，然后使用实例化后的对象调用类中的 run()方法，将执行子类重写后的 run()方法的功能而非父类方法的功能。

实例 216　美食抽象类

（实例位置：配套资源\SL\12\216）

实例说明

通过本实例可以演示抽象类不能被实例化，只能被继承。本实例首先定义抽象的父类





美食类，然后定义该类的子类面包类，最后分别对这两个类进行实例化并调用获取制作原料的方法。运行本实例，将在页面打印如图 12.14 所示的错误提示，从错误提示可知，抽象类不能被实例化。

如果将抽象类实例的代码注释掉，再次运行本实例，将出现如图 12.15 所示的结果，程序可以正常运行说明抽象类可以被继承。

Fatal error: Cannot instantiate abstract class Food in
D:\AppServ\www\MR\07\016\index.php on line **15**

图 12.14　实例抽象类的错误提示

制作面包的材料是：面粉

图 12.15　调用抽象类的子类的结果

实现过程

具体步骤如下：

（1）定义抽象美食类 Food，在该类中定义$material 属性，并通过构造方法对该属性进行初始化，然后定义 getMaterial()方法返回美食的烹制材料，代码如下：

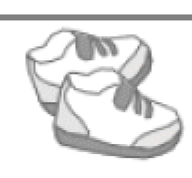
```
abstract class Food                                         //美食类
{
    private $material;                                      //烹制材料
    public function __construct ($material)                 //父类构造方法
    {
        $this->material = $material;
    }
    public function getMaterial ()                          //获得美食材料方法
    {
        return $this->material;
    }
}
```

（2）定义面包类 Bread，并使其继承自美食类 Food，然后在该类的构造方法中使用 parent 关键字调用父类的构造方法实现对其父类的初始化。该过程的实现代码如下：

```
class Bread extends Food                                    //面包类继承美食类
{
    public function __construct ($material)                 //子类构造方法
    {
        parent::__construct($material);
    }
}
```

（3）使用 require 语句在 index.php 中包含 Food.php 文件，然后使用 new 关键字对 Bread 类进行初始化，最后通过其实例的对象调用 getMaterial()方法打印出制作面包所需的材料，代码如下：

```
require 'Food.php';
$material = '面粉';
//$food = new Food($material);
```



```
//echo $food->getMaterial().'<br/>';
$bread = new Bread($material);
echo '制作面包的材料是：'.$bread->getMaterial().'<br/>';
```



Note

技术要点

抽象类与普通类的区别在于，抽象类不能被实例化，即不能用来创建对象，只能被继承。抽象类的定义方法是在 class 前加关键字 abstract，代码如下：

```
abstract class MyClass{
    //类中的属性和方法
}
```

如果对抽象类进行实例化，将在页面打印如图 12.14 所示的错误信息。

实例 217　学生类多重接口的实现

（实例位置：配套资源\SL\12\217）

实例说明

本实例主要介绍 PHP 面向对象编程中接口的定义、应用以及类如何实现多个接口。运行本实例，如图 12.16 所示，首先在表单中输入学生学号和学生姓名，然后单击“提交”按钮即可在页面中打印出学生信息。



图 12.16　打印学生信息

实现过程

具体步骤如下：

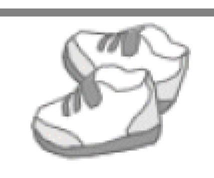
（1）定义学生 ID 接口，在接口中声明用于设置学生 ID 的 setId()方法和获得学生 ID 的 getId()方法，代码如下：

```
interface Property_Id                        //编号接口
{
    public function setId ($id);             //方法声明
    public function getId ();
}
```

（2）定义学生名称接口 Property_Name，并在接口中声明设置学生姓名的 setName()方法或获得学生姓名的 getName()方法，代码如下：

```
interface Property_Name                      //名称接口
{
    public function setName ($name);         //方法声明
    public function getName ();
}
```

（3）定义学生类 Student，使其同时实现 Property_ID 接口和 Property_Name 接口，并



实现类中所声明方法的定义，代码如下：

```
class Student implements Property_Id, Property_Name      //定义学生类，使其同时实现 Property_Id 和 Property_Name 两个接口
{
    private $id;                                  //编号属性
    private $name;                                //名称属性
    public function setId ($id)                   //实现接口中的各个方法
    {
        $this->id = $id;
    }
    public function getId ()
    {
        return $this->id;
    }
    public function setName ($name)
    {
        $this->name = $name;
    }
    public function getName ()
    {
        return $this->name;
    }
}
```



（4）建立学生属性信息录入表单，当单击表单中的“提交”按钮后将通过如下代码在页面中打印出所录入的学生编号和学生名称。

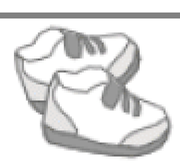
```
if(isset($_POST['id']) && $_POST['id']!=''){              //判断是否提交了表单
    require 'Student.php';                                //包含学生类
    $student = new Student();                             //对学生类进行实例化
    $student->setId($_POST['id']);                        //设定学生 ID
    $student->setName($_POST['name']);                    //设定学生名称
    echo '<font color="green">当前学生的学号是：'.$student->getId().'   姓名是：'.$student->getName().'</font>';   //打印学生信息
}
```

在上述代码中，首先判断是否已经提交了表单，如果是则首先使用 require 语句包含 Student.php 文件，然后使用 new 关键字对学生类 Student 进行实例化，并调用实例化后对象的 setXxx()方法分别为学生编号和学生姓名赋初值，最后分别调用类中的 getXxx()方法打印学生信息。

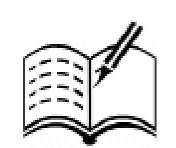
技术要点

本实例的关键技术是如何定义接口，以及如何实现一个或多个接口。在 PHP 中接口使用关键字 interface 声明，在类体内不能有属性定义，只能有方法声明，并且不能在方法中定义任何功能代码。PHP 中接口的定义方法如下：

```
interface name
{
```



```
        public function fun1();
        private function fun2($param);
    }
```



Note

对接口定义完成后，可以在定义类时使用关键字 implements 来实现一个或多个接口，所实现的多个接口名称之间使用逗号分隔，例如：

```
class MyClass implements interface1, interface2, interface3
{
    //需要实现各个接口中所声明的方法
}
```

综上可知，实现接口的类中至少应包括所实现接口中所声明的方法。

实例 218　通过继承实现多态

（实例位置：配套资源\SL\12\218）

实例说明

本实例主要通过继承实现类的多态。在制作本实例时，首先定义动物类 Animal，然后分别定义企鹅类 Penguin 和昆虫类 Insect，并在这两个子类中分别重写动物类的行走方法，最后通过这两个子类实例的对象调用其自身的行走方法将在页面打印如图 12.17 所示的内容。

```
通过继承实现多态:
企鹅可以直立行走
昆虫可以爬行
```

图 12.17　通过继承实现类的多态

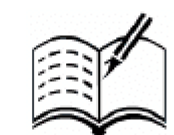
实现过程

具体步骤如下：

（1）定义抽象的动物类 Animal，并在该类中定义行走的方法 walk()，然后定义企鹅 Penguin 和昆虫类 Insect，使二者分别继承类 Animal，并在这两个类中重写 Animal 类中的 walk()方法，代码如下：

```
abstract class Animal                        //定义动物类
{
    public function walk ()                  //定义动物类中的行走方法
    {
        return '动物能行走';
    }
}
class Penguin extends Animal                 //定义企鹅类，并使其继承自动物类
{
    public function walk ()                  //重写父类中的 walk()方法
    {
        return '企鹅可以直立行走';
```





```
        }
    }
    class Insect extends Animal                    //定义昆虫类，并使其继承自动物类
    {
        public function walk ()                    //重写父类中的 walk()方法
        {
            return '昆虫可以爬行';
        }
    }
```

（2）使用 require 语句包含 Animal.php 文件，然后分别对企鹅类和昆虫类进行实例化，并分别调用这两个类中的 walk()方法，代码如下：

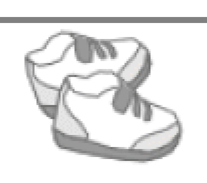
```
    require 'Animal.php';                          //包含 Animal.php 文件
    $penguin = new Penguin();                      //对企鹅类进行实例化
    echo $penguin->walk();                         //调用企鹅类的 walk()方法
    echo '<br/>';
    $insect = new Insect();                        //对昆虫类进行实例化
    echo $insect->walk();                          //调用昆虫类的 walk()方法
```

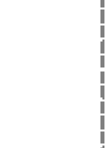
通过上述实例的运行结果可知，企鹅类和昆虫类虽然都重写了父类的 walk()方法，但在页面中打印的结果却不同，这就是通过继承实现类重载的体现。

技术要点

本实例的关键是如何通过继承来实现类的多态，在 PHP 面向对象编程中，可以首先定义一个抽象的父类，然后再定义多个子类来继承该父类，在子类中可以通过方法重载的方式来重写父类中的功能，这样就实现了类的多态。其示例代码如下：

```
    abstract class Parent                          //定义抽象的父类 Parent
    {
        public function fun ()                     //定义父类的 fun()方法
        {
            return '父类 fun()方法的实现';
        }
    }
    class Child1 extends Parent                    //定义子类 Child1，使其继承 Parent
    {
        public function fun ()                     //在 Child1 中重写 Parent 类中的 fun()方法
        {
            return 'Child1 中 fun()方法的实现';
        }
    }
    class Child2 extends Parent                    //定义子类 Child2，使其继承 Parent
    {
        public function fun ()                     //在 Child2 中重写 Parent 类中的 fun()方法
        {
            return 'Child2 中 fun()方法的实现';
        }
    }
```





Note

实例 219 通过接口实现多态

（实例位置：配套资源\SL\12\219）

实例说明

本实例主要介绍如何通过接口来表现类的多态性。制作本实例时，首先定义动物接口 Animal，然后分别定义企鹅类和昆虫类来实现这个接口，并实现接口中所声明的 walk()方法，运行本实例，如图 12.18 所示。

通过接口实现多态：

企鹅可以直立行走

昆虫可以爬行

图 12.18 通过接口实现类的多态

实现过程

具体步骤如下：

（1）定义动物类接口 Animal，并在该接口中声明 walk()方法，然后定义企鹅类 Animal 和昆虫类 Insect，使这两个类分别实现 Animal 接口，代码如下：

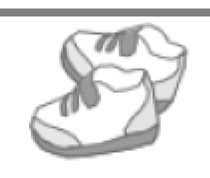
```
interface Animal                              //定义动物接口
{
    public function walk ();                  //声明行走的方法
}
class Penguin implements Animal               //定义企鹅类，并使其实现动物接口
{
    public function walk ()                   //重写父类中的 walk()方法
    {
        return '企鹅可以直立行走';
    }
}
class Insect implements Animal                //定义昆虫类，并使其实现动物接口
{
    public function walk ()                   //重写父类中的 walk()方法
    {
        return '昆虫可以爬行';
    }
}
```

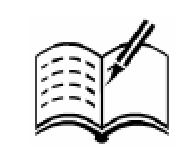
（2）分别对企鹅类和昆虫类进行实例化，然后使用实例化的对象调用类中的 walk()方法，代码如下：

```
require 'Animal.php';                         //包含 Animal.php 文件
$penguin = new Penguin();                     //对企鹅类进行实例化
echo $penguin->walk();                        //调用企鹅类的 walk()方法
echo '<br/>';
$insect = new Insect();                       //对昆虫类进行实例化
echo $insect->walk();                         //调用昆虫类的 walk()方法
```

技术要点

在继承中通过方法重写方式可以体现类的多态性，通过接口的方式同样可以体现类的





多态性，其示例代码如下：

```
interface A                                      //定义接口 A
{
    public function fun ();                      //声明接口中的 fun()方法
}
class Class1 implements A                        //定义类 Class1 使其实现接口 A
{
    public function fun ()                       //实现接口 A 中的 fun()方法
    {
        return '在类 Class1 中实现接口 A 中的 fun()方法';
    }
}
class Class2 implements A
{
    public function fun ()                       //实现接口 A 中的 fun()方法
    {
        return '在类 Class2 中实现接口 A 中的 fun()方法';
    }
}
```

上述代码中定义接口 A，并在接口中声明 fun()方法，然后分别定义类 Class1 和类 Class2，使这两个类都实现接口 A，并实现接口 A 中的 fun()方法。从示例代码中可知，类 Class1 和类 Class2 虽然都实现了接口 A 中所声明的 fun()方法，但这两个方法实现的功能却不同，从而体现类的多态性。

实例 220　使用 final 关键字防止类被继承

（实例位置：配套资源\SL\12\220）

实例说明

本实例主要讲解 final 关键字的用法。首先定义水果类 Fruit，然后定义 final 型的苹果类 Apple，使之继承自水果类，通过 Apple 类即可实现对苹果属性的设置。运行本实例，如图 12.19 所示，首先在文本框中输入苹果的颜色和形状属性，然后单击“提交”按钮即可在页面中打印出苹果的属性信息。如果再定义一个类，并使之继承 final 型的 Apple 类，则会在页面中打印出如图 12.20 所示的错误提示。

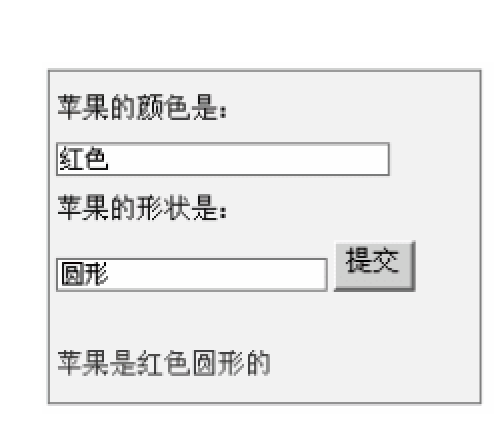
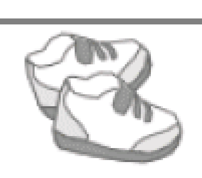


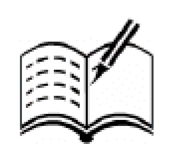
图 12.19　打印苹果的属性

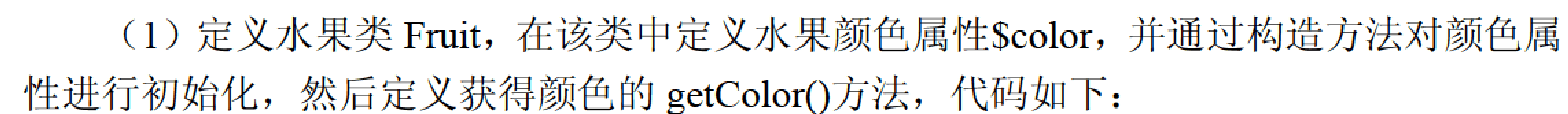


图 12.20　继承 final 型水果类的错误提示



实现过程





具体步骤如下：

（1）定义水果类 Fruit，在该类中定义水果颜色属性$color，并通过构造方法对颜色属性进行初始化，然后定义获得颜色的 getColor()方法，代码如下：

```
class Fruit                                    //定义水果类
{
    private $color;                            //定义颜色属性
    public function __construct ($color)       //通过构造方法对颜色属性进行初始化
    {
        $this->color = $color;
    }
    public function getColor ()                //获得水果颜色的方法
    {
        return $this->color;
    }
}
```

（2）定义 final 型的苹果类 Apple，并使其继承水果类 Fruit，在水果类中定义苹果的形状属性$shape，通过构造方法对苹果类进行初始化，最后定义 getShape()方法，返回水果的形状，代码如下：

```
final class Apple extends Fruit                //定义 final 型的苹果类，使之继承子水果类
{
    private $shape;                            //定义形状属性
    public function __construct ($color, $shape)   //构造函数
    {
        parent::__construct($color);           //调用父类构造方法
        $this->shape = $shape;                 //对形状属性进行初始化
    }
    public function getShape ()                //获得苹果的形状方法
    {
        return $this->shape;
    }
}
/*
class Test extends Apple {          //定义一个 Test 测试类，使之继承 final 型的 Apple 类

}
*/
```

上述代码中，用多行注释的部分用于测试被关键字 final 修饰的类是否可以被继承。

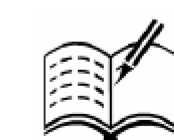
（3）建立苹果属性录入表单，当用户单击表单中的“提交”按钮后，将通过如下代码打印出苹果的属性。

```
if(isset($_POST['color']) && $_POST['color']!=''){
    require 'Fruit.php';
    $apple = new Apple($_POST['color'], $_POST['shape']);
```



```
        echo '<font color="red">苹果是'.$apple->getColor().$apple->getShape().'的</font>';
    }
```

上述代码中，首先判断用户是否已经提交了表单，如果是则首先使用 require 语句包含 Fruit.php 文件，然后对苹果类进行实例化，最后分别通过 getColor()方法和 getShape()方法打印出苹果的属性。



技术要点

通过本实例的运行结果可知，使用 final 关键字修饰的类只能被实例化，不能被继承，如果继承了 final 型的类，将在页面中打印如下错误提示。

```
    Fatal error: Class Test may not inherit from final class (Apple) in D:\AppServ\www\MR\07\020\Fruit.php on line 30
```

实例 221　使用 static 关键字定义类的静态成员

（实例位置：配套资源\SL\12\221）

实例说明

通过本实例主要讲解 PHP 面向对象编程方式中的 static 关键字的用法，运行本实例，如图 12.21 所示，首先在图中的文本框中输入要计算的数字，同时在计算类型列表框中选择计算类型，然后单击“求值”按钮，即可在页面中打印出计算结果。这里要注意，在进行除运算时，如果除数为 0，则应给出错误提示，如图 12.22 所示。

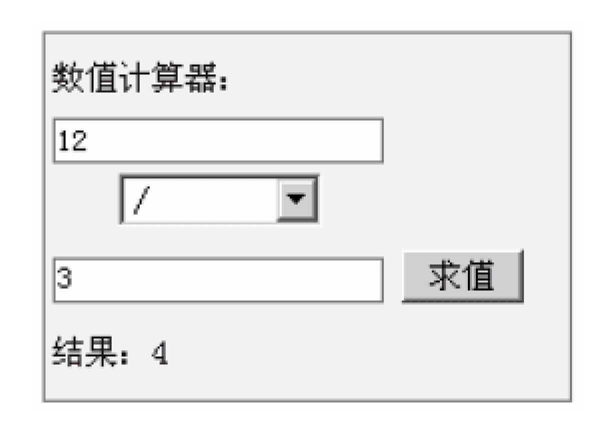


图 12.21　数值计算器

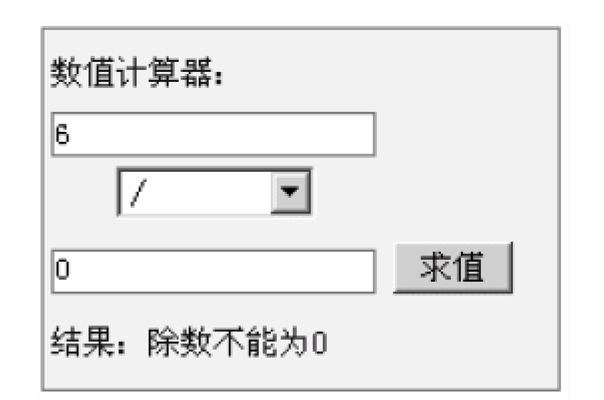


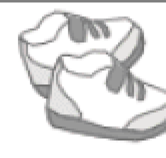
图 12.22　除数为 0 时的提示信息

实现过程

具体步骤如下：

（1）定义数值计算类 Math，并在该类中定义用于进行加、减、乘、除运算的静态方法，在进行求除运算中通过 try/catch 语句进行除数为 0 时的异常处理。实现该过程的代码如下：

```
class Math                                              //定义数值计算类
{
    public static function add ($num1, $num2)           //相加的方法
    {
        return $num1 + $num2;
    }
    public static function sub ($num1, $num2)           //相减的方法
    {
```

```
            return $num1 - $num2;
        }
        public static function multi ($num1, $num2)            //相乘的方法
        {
            return $num1 * $num2;
        }
        public static function div ($num1, $num2)              //相除的方法
        {
            try {
                if ($num2 == 0) {                              //如果除数为 0，则抛出异常
                    throw new Exception('除数不能为 0');
                } else {
                    return $num1 / $num2;
                }
            } catch (Exception $e) {
                return $e->getMessage();                       //如果除数为 0，则给出错误提示
            }
        }
    }
```

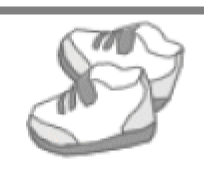
（2）定义数值录入表单，当在表单中录入数字，并单击表单的“求值”按钮后，将通过如下代码计算出表单中所定义的算数运算的结果。

```
    if (isset($_POST['num1']) && trim($_POST['num1']) != '') {      //判断是否提交了表单
        require 'Math.php';                                          //包含 Math.php 文件
        $num1 = $_POST['num1'];                                      //获得提交的数字
        $num2 = $_POST['num2'];
        switch ($_POST['type']) {
          case '+':                                                  //进行加运算
                $result = Math::add($num1, $num2);
                break;
            case '-':                                                //进行减运算
                $result = Math::sub($num1, $num2);
                break;
            case '*':                                                //进行乘运算
                $result = Math::multi($num1, $num2);
                break;
            case '/':                                                //进行除运算
                $result = Math::div($num1, $num2);
                break;
        }
        echo '结果：' . $result;                                      //打印计算结果
    }
```

上述代码中，首先判断表单是否已经提交，如果是则使用 switch 语句判断要进行计算的类型，然后通过 Math 类直接调用其中的静态方法进行计算。

技术要点

在 PHP 面向对象编程方式中，使用 static 关键字修饰的属性称为静态属性，使用 static 关键字修饰的方法称为静态方法。通过类实例的对象可以调用类中公有方法，而静态成员的



特点是在不对类进行实例化的前提下，直接通过类名即可对其进行调用，其调用格式如下：

```
类名::公有静态成员
```

实例 222　使用 clone 关键字实现对象的克隆

（实例位置：配套资源\SL\12\222）

实例说明

本实例主要讲解 PHP5.0 面向对象编程中 clone 关键字的使用方法，运行本实例，分别如图 12.23 和图 12.24 所示，其中图 12.23 中的输出结果是使用 clone 关键字克隆对象后，分别用原来对象和克隆的对象对类中方法进行重新赋值并调用的结果，而图 12.24 为使用等号赋值的方式来产生一个新对象后，然后再分别使用新旧对象实现对类中的方法进行调用的结果。

```
羊的颜色是白色
羊的颜色是灰色
```

图 12.23　使用 clone 关键字传递对象

```
羊的颜色是白色
羊的颜色是白色
```

图 12.24　使用等号传递对象

实现过程

具体步骤如下：

（1）建立羊类 Sheep，并在该类中定义羊的颜色属性$color，同时定义用来设置颜色的 setColor()方法和获得颜色的 getColor()方法，代码如下：

```
class Sheep
{
    private $color;                                    //颜色属性
    public function setColor ($color)                  //设置颜色的方法
    {
        $this->color = $color;
    }
    public function getColor ()                        //获得颜色的方法
    {
        return '羊的颜色是' . $this->color;
    }
}
```

（2）对 Sheep 类进行实例化，然后分别用等号赋值和 clone 的方法测试对对象进行更改的影响结果，代码如下：

```
require 'Sheep.php';                    //包含 Sheep.php 文件
$sheep = new Sheep();                   //对 Sheep 类进行实例化
/*
$sheep->setColor('白色');               //设置羊的颜色
echo $sheep->getColor();                //打印羊的颜色
$sheep1 = $sheep;                       //将$sheep 对象赋值给新对象$sheep1
```




```
echo '<br/>';
$sheep1->setColor('灰色');                  //通过新对象$sheep1 调用设置颜色的方法
echo $sheep->getColor();                    //打印羊的颜色
*/
$sheep->setColor('白色');                   //设置羊的颜色
echo $sheep->getColor();                    //打印羊的颜色
$sheep1 = clone $sheep;                     //克隆$sheep 对象，产生一个新的$sheep1 对象
echo '<br/>';
$sheep1->setColor('灰色');               //通过新克隆的$sheep1 对象调用 Sheep 类中的 setColor()方法
echo $sheep->getColor();                    //打印羊的颜色
```

技术要点

在 PHP5.0 以后的版本中，使用 clone 关键字实现对对象的克隆，该关键字的语法格式如下：

```
$obj_new = clone $obj_old;
```

初学者可能会问，直接用等号不就可以将当前对象赋给一个其他对象么，为什么还要用 clone 关键字呢？这主要是因为，在 PHP5.0 中对象被存储于独立的结构 Object Store 中，而不像其他一般变量存储于 Zval 中，在 Zval 中仅存储对象地址的引用，而不是内容，当赋值一个变量或者将一个变量传递给一个函数时，就不再复制数据了，所以对新变量的更改不会导致原变量内容的更改，而采用 Object Store 存储对象时，使用等号产生一个新对象后，对新对象的更改会导致原对象的更改，而使用 clone 关键字所克隆出的对象，新对象的更改并不会导致原对象的更改。

指点迷津：

使用 clone 关键字克隆一个对象，就如同克隆一个人，打一下新克隆出的人，被克隆的人并不会感到疼痛，其中的道理是一样的。

实例 223　使用__set()方法为类中未经定义的属性赋值

（实例位置：配套资源\SL\12\223）

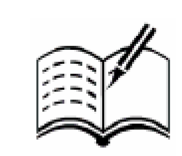
实例说明

通过本实例主要讲解如何使用 PHP 中的__set()方法存取类中未声明的属性。运行本实例，如图 12.25 所示，在页面中以表格的形式列出各图书的详细信息，其中图书的“备注”属性是使用__set()方法所赋的值。

书　名	页　码	作　者	价　格	备　注
《PHP从基础到**》	650	小张、小潘、小王	58	备注
《PHP函数**》	800	小潘、小王	80	备注
《PHP范例**》	700	小李、小懂	85	备注
《PHP实战**》	750	小郭、小刘	75	备注

图 12.25　图书信息列表





实现过程

具体步骤如下：

（1）定义图书类 Book，在类中定义图书的名称、页码、作者和价格等属性，并分别定义这些属性的 setXxx()和 getXxx()方法，然后在类中声明__set()方法，使用该方法为类中未声明的属性赋初值。实现该过程的代码如下：

```
class Book                                          //定义图书类
{
    private $name;                                  //书名
    private $page;                                  //页码
    private $writer;                                //作者
    private $price;                                 //价格
    private $other;                                 //其他信息
  public function setName ($name)                   //设置书名
    {
        $this->name = $name;
    }
    public function getName ()                      //获得书名
    {
        return $this->name;
    }
    public function setPage ($page)                 //设置页码
    {
        $this->page = $page;
    }
    public function getPage ()                      //获得页码
    {
        return $this->page;
    }
    public function setWriter ($writer)             //设置作者
    {
        $this->writer = $writer;
    }
    public function getWriter ()                    //获得作者
    {
        return $this->writer;
    }
    public function setPrice ($price)               //设置价格
    {
        $this->price = $price;
    }
    public function getPrice ()                     //获得价格
    {
        return $this->price;
    }
    public function __set ($name, $value)
    {
        $this->other = $value;
```



```
        }
        public function getOther ()
        {
            return $this->other;
        }
}
```



（2）建立图书信息二维数组，用于保存图书信息，代码如下：

```
require 'Book.php';
$arrayBook = array(                          //图书信息数组，用于模拟图书信息数据表
    array('name'=>'《PHP 从基础到**》', 'page'=>'650', 'writer'=>'小张、小潘、小王','price'=>'58'),
    array('name'=>'《PHP 函数**》', 'page'=>'800', 'writer'=>'小潘、小王','price'=>'80'),
    array('name'=>'《PHP 实例**》', 'page'=>'700', 'writer'=>'小李、小懂','price'=>'85'),
    array('name'=>'《PHP 实战**》', 'page'=>'750', 'writer'=>'小郭、小刘','price'=>'75')
);
```

（3）使用 foreach 语句通过循环输出图书信息。在 foreach 循环体内部，首先对 Book 类进行实例化，然后分别调用类中的 setXxx()方法为类中已声明的属性赋初值，同时为类中的未声明的备注属性赋值，最后使用 getXxx()方法打印出图书的信息。实现该过程的代码如下：

```
<?php
require 'db.php';
foreach ($arrayBook as $key => $aBook) {
    $book = new Book();
    $book->setName($aBook['name']);
    $book->setPage($aBook['page']);
    $book->setWriter($aBook['writer']);
    $book->setPrice($aBook['price']);
    $book->bz = '备注';
?>
<div style="width:100%; <?php if($key < count($arrayBook)-1){?>border-bottom:1px solid #0463BD;
<?php } ?> clear:both;">
    <div style="width:160px; height:22px; line-height:22px; text-align:left; float:left;">
      <?php echo $book->getName()?>
    </div>
    <div style="width:160px; height:22px; line-height:22px; border-left:1px solid #0463BD; float:left;">
    <?php echo $book->getPage()?>
    </div>
    <div style="width:160px; height:22px; line-height:22px; border-left:1px solid #0463BD; float:left;">
    <?php echo $book->getWriter()?>
    </div>
    <div style="width:160px; height:22px; line-height:22px; border-left:1px solid #0463BD; float:left;">
    <?php echo $book->getPrice()?>
    </div>
    <div  style="width:156px; height:22px; line-height:22px; border-left:1px solid #0463BD; float:left;">
    <?php echo $book->getOther()?>
    </div>
</div>
<?php }?>
```





Note

技术要点

魔术方法__set()的作用是存取类中未声明的属性，该方法必须接收两个参数，分别用来表示类中未声明的属性名和属性值。__set()方法的语法格式如下：

```
function __set($name, $value)
{
    //可以使用$name 和$value 在方法中存取类中未定义的属性名和数值
}
```

上述__set()方法中的参数$name 表示类中未声明属性的名称，$value 表示类中未声明属性的值。

实例 224　使用__get()方法获取未声明属性的名称

（实例位置：配套资源\SL\12\224）

实例说明

本实例主要应用__get()方法获得类中未定义属性的值。运行本实例，如图 12.26 所示，页面中打印的内容是通过调用苹果类的属性打印的结果，而弹出的提示对话框是因为没有在类中定义$produceArea 属性而通过__get()方法弹出的。



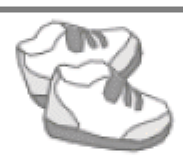
图 12.26　类中未定义属性提示框

实现过程

具体步骤如下：

（1）定义苹果类 Apple，在该类中分别定义苹果颜色、形状和重量等属性，并使用构造方法对这些属性进行初始化，然后定义用于获得苹果属性信息的 getProperty()方法，最后定义__get()方法用于弹出未定义属性名称的对话框。实现该过程的代码如下：

```
class Apple
{
    private $color;                                          //颜色
    private $shape;                                          //形状
    private $weight;                                         //重量
    public function __construct ($color, $shape, $weight)   //构造方法
    {
        $this->color = $color;
        $this->shape = $shape;
        $this->weight = $weight;
    }
    public function getProperty ()
    {
        return '这个苹果重' . $this->weight . '，是' . $this->color . '，' . $this->shape . '的！';
    }
```




```
            public function __get ($name)
            {                                                //使用__get()方法弹出未定义属性的提示
                echo '<script>alert("在类中未定义属性' . $name . '！ ");</script>';
            }
        }
```

（2）使用 new 关键字对苹果类 Apple 进行实例化，然后使用实例化后的对象调用苹果类中的 getProperty()方法打印苹果的属性信息，最后使用苹果对象调用类中未定义的属性。实现该过程的代码如下：

```
        require 'Apple.php';                             //使用 require 语句包含 Apple.php 文件
        $apple = new Apple('红色', '圆形', '0.4kg');     //使用 new 关键字对 Apple 类进行实例化
        echo $apple->getProperty();                      //调用类中的 getProperty()属性
        echo $apple->produceArea;
```

技术要点

魔术方法__get()用于获得类中不存在属性的名称，该方法必须带有一个表示未定义属性名称的参数。__get()方法使用格式如下：

```
        function __get($name)
        {
            //在方法内部可以使用$name 的值，该值即为未定义方法的名称
        }
```

实例 225　使用__call()方法打印类中未定义方法的信息

（实例位置：配套资源\SL\12\225）

实例说明

通过本实例主要讲解__call()魔术方法的使用。运行本实例，如图 12.27 所示，将在页面中打印出图书的信息和调用未定义方法的名称。其中图书的信息是图书类调用类中的 getProperty()方法所返回的结果，而未定义 getInfo 方法的提示是通过类中的__call()方法的打印结果。

《PHP范例**》的价格是85元

getInfo方法未定义

图 12.27　类中未定义属性提示框

实现过程

具体步骤如下：

（1）定义图书类 Book，首先在该类中定义$name 和$price 属性，并使用构造方法对这两个参数进行实例化，然后定义 getProperty()方法返回图书属性，最后定义__call()方法打印未定义方法的名称。实现该过程的代码如下：

```
        class Book                                       //图书类
        {
            private $name;                               //书名
```






```
        private $price;                                   //价格
        public function __construct ($name, $price)       //构造方法
        {
            $this->name = $name;
            $this->price = $price;
        }
        public function __call ($name, $arguments)        //__call()方法
        {
            echo $name . '方法未定义';
        }
        public function getProperty ()                    //获得图书信息的方法
        {
            return $this->name . '的价格是' . $this->price . '元';
        }
    }
```

（2）使用 new 关键字对 Book 类进行实例化，然后使用实例化的对象调用类中的 getProperty()方法打印图书信息，最后使用实例后的对象调用类中未定义的 getInfo()方法。实现该过程的代码如下：

```
require 'Book.php';                               //包含 Book.php 文件
$bookName = '《PHP 实例**》';                     //书名
$price = '85';                                    //价格
$book = new Book($bookName, $price);              //对 Book 类进行实例化
echo $book->getProperty() . '<br/>';              //打印图书属性
$book->getInfo($bookName, $price);                //调用类中未定义的 getInfo()方法
```

技术要点

魔术方法__call()的作用是在调用类中一个不存在或不可见的方法时将执行该方法，__call()方法必须接收两个参数，用来存放视图调用的方法名及其参数。该方法的语法格式如下：

```
function __call($name, $arguments)
{

}
```

上述实例代码中的参数$name 用于表示类中未定义方法名称，$arguments 用于表示未定义方法的参数所组成数组。

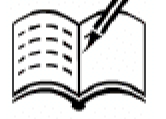
实例 226　使用__toString()方法将类的实例转化为字符串

（实例位置：配套资源\SL\12\226）

实例说明

本实例主要讲解如何通过__toString()魔术方法将类实例的对象转换为字符串，运行本实例，如图 12.28 所示，在页面中打印出圆的半径和该半径对应的圆面积，其中圆的面积





是直接打印圆类实例对象的结果。



图 12.28　打印圆的半径和面积

实现过程

具体步骤如下：

（1）创建圆类 Circle，在圆类中定义圆半径私有属性$radius，并使用构造方法对该属性进行初始化，然后定义__toString()魔术方法，使用该方法返回圆的面积值，代码如下：

```
class Circle                                                    //定义圆类
{
    private $radius;                                            //圆半径属性
    public function __construct ($radius)                       //构造方法
    {
        $this->radius = $radius;
    }
    public function __toString ()                               //定义__toString()方法
    {
        return "圆的面积是：" . (string) number_format((pi() * pow($this->radius, 2)), 2);
    }
}
```

（2）使用 require 语句包含 Circle.php 文件，然后设定圆的半径为 2，并使用 echo 语句输出圆的半径和圆类实例的对象，代码如下：

```
require 'Circle.php';                                           //包含 CircleArea.php 文件
$radius = 2;                                                    //圆半径
echo '圆的半径是：'.$radius.'<br/>';                            //打印圆半径的值
echo new Circle($radius);                                       //打印圆面积
```

技术要点

魔术方法__toString()的作用是将类实例的对象转换为字符串，该方法的语法格式如下：

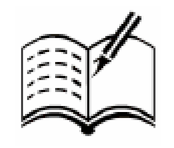
```
function __toString(){
    return  字符串;
}
```

该方法的返回值必须为字符串类型，如果在类中未定义该方法，当直接输出某一个类实例的对象时，将在页面中打印如下错误提示：

```
Catchable fatal error: Object of class Test could not be converted to string in D:\AppServ\www\MR\07\027\t.php on line 3
```

当在类中定义__toString()方法后，直接输出类实例后的对象将在页面中打印__toString()方法所返回的字符串。





实例 227　使用__isset()方法提示未定义属性信息

（实例位置：配套资源\SL\12\227）

实例说明

运行本实例，如图 12.29 所示，在页面中打印出未定义属性的提示信息以及“小明”的年龄和体重信息，其中类中未定义属性 sex 的提示信息是通过魔术方法__isset()打印的。

```
在类中未定义属性sex

小明的年龄是：12岁，体重是：45公斤
```

图 12.29　打印人的属性信息

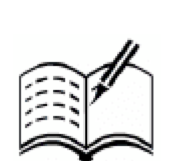
实现过程

具体步骤如下：

（1）定义人类 Person，在该类中定义人的名称、年龄和体重等属性，并定义与这些属性相对应的 setXxx()方法和 getXxx()方法，代码如下：

```
class Person
{
    private $name;                                    //名称
    private $age;                                     //年龄
    private $weight;                                  //体重
    public function setName ($name)                   //设置名称
    {
        $this->name = $name;
    }
    public function getName ()                        //获得名称
    {
        return $this->name;
    }
    public function setAge ($age)                     //设置年龄
    {
        $this->age = $age;
    }
    public function getAge ()                         //获得年龄
    {
        return $this->age;
  }
    public function setWeight ($weight)               //设置体重
    {
        $this->weight = $weight;
    }
    public function getWeight ()                      //获得体重
```






```
        {
            return $this->weight;
        }
        public function __isset ($name)                    //定义__isset()方法
        {
            echo '在类中未定义属性' . $name;
        }
}
```

（2）使用 new 关键字对 Person 类进行实例化，然后分别调用该类中相应的 setXxx()方法为类中的属性赋初值，并使用函数 isset()判断类中未定义的 sex 属性是否被设置。实现该过程的代码如下：

```
require 'Person.php';                              //包含 Person.php 文件
$person = new Person();                            //实例 Person 类
$person->setName('小明');                          //设置名称
$person->setAge('12');                             //设置年龄
$person->setWeight('45 公斤');                     //设置体重
isset($person->sex);                               //判断是否在类中定义了 sex 属性
echo '<br/>';                                      //换行
echo $person->getName().'的年龄是：'.$person->getAge().'岁，体重是：'.$person->getWeight() ;
//打印信息
```

技术要点

魔术方法__isset()的作用是当使用 isset()函数判断类中未定义的属性是否被设置时所调用的。该方法的使用格式如下：

```
function __isset ($name)
{

}
```

魔术方法__isset()必须包含一个用于代表类中未定义属性的名称参数。

实例 228　使用单例模式制作数据库管理类

（实例位置：配套资源\SL\12\228）

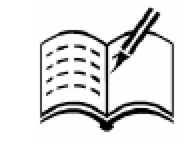
实例说明

本实例主要讲解如何使用单例模式制作一个基于 MySQL 数据库的数据库管理类。运行本实例，如图 12.30 所示，首先在用户登录文本框中输入用户名和密码，然后单击表单中的“登录”按钮，如果用户名和密码经后台代码验证正确则当前页面将跳转到登录成功信息提示页面，如果错误则在页面打印错误提示，在上述过程中，对数据库管理模块在技术上采用 PDO 技术，设计上是采用单例模式进行编码的。





图 12.30　用户登录表单

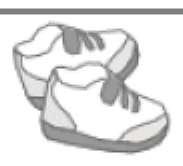


实现过程

具体步骤如下：

（1）建立数据库连接类 Db，在类中定义用于保存当前类实例的静态成员变量$instance，并定义静态方法 getInstace()。在该方法内部将当前类实例的对象赋给静态成员变量$instance 并返回，然后定义私有的构造方法和私有的克隆方法，这样就构建了单例模式的基本框架，接下来创建 getConnId()方法。该方法用于返回使用 PDO 技术连接数据库后的句柄，然后创建数据库查询方法 query()实现对数据库的查询操作。实现该过程的代码如下：

```
class Db                                                          //数据库类
{
    private static $instance;                                     //保存当前类实例后的对象
    public static function getInstance ()                         //静态方法，返回类实例
    {
        if (null == self::$instance) {                            //如果未定义
            self::$instance = new Db();
        }
        return self::$instance;
    }
    private function __construct ()                               //私有构造方法
    {}
    private function __clone ()                                   //私有克隆方法
    {}
    private function getConnId ()                                 //获得数据库连接 ID 的方法
    {
        $config = parse_ini_file(dirname(__FILE__) . '/config.ini');        //解析配置文件
        $dsn = 'mysql:host=' . $config['host'] . ';dbname=' . $config['dbname'];   //配置数据源名称
        $pdo = new PDO($dsn, $config['username'], $config['password']);      //实例 PDO 对象
        $pdo->query('set names ' . $config['charset']);                      //设置字符集
        return $pdo;                                                          //返回 PDO 对象
    }
    public function query ($sql)                                              //查询方法
    {
        $pdo = $this->getConnId();                                            //获得 PDO 对象
```




```
                $sqlType = trim(substr($sql, 0, 6));                              //获得查询语句类型
                if ($sqlType == 'update' || $sqlType == 'delete' || $sqlType == 'insert') {
                    return $pdo->query($sql);                                     //如果是 update、delete
或 insert 语句则返回 SQL 语句执行结果
                } elseif ($sqlType == 'select') {                                 //如果是 select 语句
                    $tmpArray = array();                                          //定义二维数组
                    $results = $pdo->query($sql);                                 //执行 select 语句
                    foreach ($results as $result) {                             //将结果保存到二维数组中
                        array_push($tmpArray, $result);
                    }
                    return $tmpArray;                                             //返回结果
                } else {
                    return null;                  //不是 update、delete、insert 或 select 语句，则返回 null
                }
            }
        }
```

（2）建立用户登录表单，其实现过程请详见本书附带源码配套资源。当用户在表单中录入完成用户名和密码，单击“提交”按钮后将通过如下代码验证用户的登录信息是否正确。

```
        if(isset($_POST['username']) && trim($_POST['username'])!='')
        {
            require_once 'Db.php';
            $username = trim($_POST['username']);
            $password = trim(md5($_POST['password']));
            $db = Db::getInstance();
            $arrayUser = $db->query("select id from tb_user where username='".$username."' and password
='".$password."'");
            if(count($arrayUser)>0){
                $_SESSION['loginUsername'] =   $username;
                echo '<script>window.location.href="success.php";</script>';
            }else {
                echo '<div style="width:300px; height:30px; line-height:30px; border:1px solid #E59B04;
background-color:#FCF2E0; color:#FF0000;">用户名或密码输入有误</div>';
            }
        }
```

上述代码首先判断用户是否提交了表单，如果是则通过 Db 类的 getInstance()方法获得类的实例，并通过该对象调用类中的 query()方法执行查询，根据查询结果的记录数来判断用户输入的用户名和密码是否正确，如果记录数大于 1 则说明用户登录信息有效。

技术要点

单例模式是指在程序应用范围内只对指定的类创建一个实例，也就是说该模式包含的对象只有一个，就是单例本身。PHP 使用单例模式所设计的类通常应满足如下要求：

☑ 单例模式的类通常拥有一个私有的构造方法和私有的克隆方法，这样可以确保用户无法通过创建对象或者克隆的方式对其进行实例化。

☑ 单例模式的类包含一个静态的用于保存当前类实例的成员变量以及一个静态方法，该静态方法负责对其本身进行实例化，然后将实例化后的对象保存到类中所



声明的静态变量中，从而确保一个对象被创建。

PHP 的单例模式的基本形式如下：

```
class Singleton
{
    private static $instance;                       //保存当前类实例后的对象
    public static function getInstance ()           //静态方法，返回类实例
    {
        if (null == self::$instance) {              //如果未定义
            self::$instance = new Db();
        }
        return self::$instance;                     //返回当前对象
    }
    private function __construct ()                 //私有构造方法
    {}
    private function __clone ()                     //私有克隆方法
    {}
    //类的其他方法
}
//获得类的对象
$singleton = Singleton::getInstance();
```

通过上述代码可知，在类外部通过类名直接调用类中的用于返回当前类实例的静态方法即可获得当前类实例。

实例 229　使用策略模式打印客户端浏览器类型

（实例位置：配套资源\SL\12\229）

实例说明

本实例主要讲解在 PHP 面向对象编程中策略模式的应用方法。运行本实例，分别如图 12.31 和图 12.32 所示，其中图 12.31 是在 IE 下运行的结果，图 12.32 为在 Firefox 浏览器下运行的结果，从运行结果可知，本实例会根据不同浏览器的类型在页面中打印相应浏览器的名称。

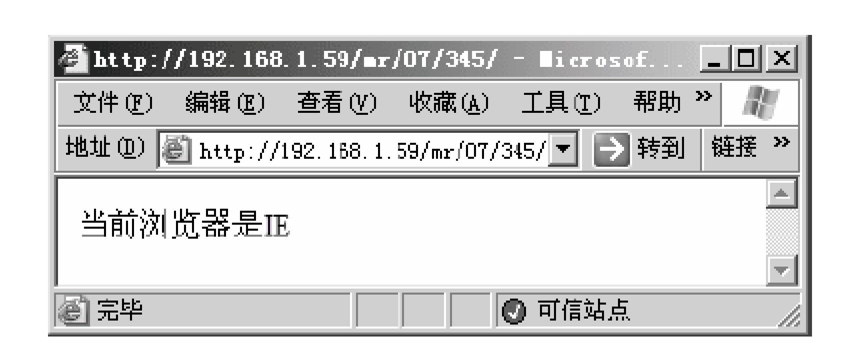


图 12.31　打印浏览器的类型为 IE



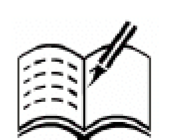
图 12.32　打印浏览器的类型是 Firefox

实现过程

具体步骤如下：

（1）定义抽象的 Browser 类，并在该类中声明抽象方法 getType()，然后定义子类 Ie 和 Fox，使这两个类分别继承基类 Browser，然后在类体内完成对 getType()类的定义，代





码如下：

```
abstract class Browser                                  //定义抽象类 Browser
{
    abstract public function getType ();
}
class Ie extends Browser                                //定义 IE 类
{
    public function getType ()                          //获得类型方法
    {
        return 'IE';
    }
}
class Fox extends Browser                               //定义 Fox 类
{
    public function getType ()                          //获得类型方法
    {
        return 'Firefox';
    }
}
```

（2）使用 header()函数设置页面编码，并使用 require 语句包含 Browser.php 文件，然后定义 getBrowserType()方法获得浏览器类型，代码如下：

```
header('content-type:text/html; charset=utf-8');        //定义页面编码
require 'Browser.php';                                  //包含 Browser.php 文件
function getBrowserType ()                              //定义获得浏览器类型的方法
{
    if (strstr($_SERVER['HTTP_USER_AGENT'], "IE")) {    //判断是否为 IE 浏览器
        $b = new Ie();                                  //实例 Ie 类
    } else {
        $b = new Fox();                                 //实例 Fox 类
    }
    return $b->getType();                               //返回浏览器类型
}
echo '当前浏览器是' . getBrowserType();                    //执行 getBrowserType()方法
```

技术要点

策略模式是指程序中涉及决策控制的一种模式，它通常定义一个抽象基类，然后根据情况的不同创建不同的子类，其中一个简单的策略模式框架如下：

```
abstract class Strategy                   //定义抽象的基类
{
    abstract public function fun ();      //定义抽象方法 fun()
}
class Child1 extends Strategy             //定义子类 Child1，使其继承基类 Strategy
{
    public function fun ()                //实现 fun()方法的功能
    {
        //Child1 中 fun()方法所实现的功能
    }
```



```
}
class Child2 extends Strategy                    //定义子类 Child2，使其继承基类 Strategy
{
    public function fun ()                       //实现 fun()方法的功能
    {
        //Child2 中 fun()方法所实现的功能
    }
}
if(条件 1==true){                                //如果条件 1 为真
    $obj = new Child1();                         //生成子类 Child1 的实例对象
}else {
    $obj = new Child2();                         //生成子类 Child2 的实例对象
}
$obj->fun();
```



上述示例代码中首先定义抽象基类 Strategy，并在类中定义抽象方法 fun()，然后分别定义类 Child1 和 Child2，使这两个类分别继承基类 Strategy，在这两个子类的内部实现 fun()的功能，完成以上类的定义后，就可以根据不同的条件对不同的子类进行实例化。

实例 230　使用工厂模式设置用户访问权限

（实例位置：配套资源\SL\12\230）

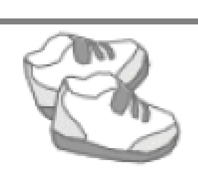
实例说明

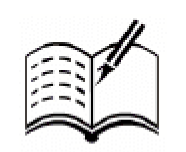
本实例主要讲解如何使用工厂模式对用户权限进行管理。运行本实例，如图 12.33 所示，在页面中将打印出不同用户的类型及各项操作的访问权限。制作本实例时，首先创建抽象的用户权限类，然后分别定义浏览、添加、更改和删除等子类，使之继承基类用户权限类，然后根据不同用户的权限，重写用户权限类中的方法，最后创建用户权限工厂来根据不同用户类别来返回相应的对象。



用户名	用户类别	浏览	添加	更改	删除
大麦	访客	是	否	否	否
SOFT	会员	是	是	否	否
春泉	访客	是	否	否	否
金星	访客	是	否	否	否
360	会员	是	是	否	否
MR	管理员	是	是	是	是

图 12.33　电子相册用户权限列表





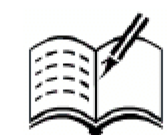
实现过程

具体步骤如下：

（1）定义用户权限基类 UserPermission，然后在该基类中定义 select()、add()、edit()和 delete()等方法，然后分别定义子类 GuestPermission、MemberPermission 和 AdminPermission，并使这些类分别继承基类 UserPermission，同时根据不同类的功能对基类中的方法进行重写，最后定义用户权限工厂 UserPermissionFactory 来根据不同用户的类别对相应的子类进行实例化并返回。实现该过程的代码如下：

```
abstract class UserPermission                       //定义抽象的用户权限类
{
    public function select ()                       //定义查询方法
    {
        return true;
    }
    public function add ()                          //定义添加方法
    {
        return false;
    }
    public function edit ()                         //定义编辑方法
    {
        return false;
    }
    public function delete ()                       //定义删除方法
    {
        return false;
    }
}
class GuestPermission extends UserPermission        //定义浏览者权限类
{}
class MemberPermission extends UserPermission       //定义会员权限类
{
    public function add ()                          //重写添加方法
    {
        return true;
    }
}
class AdminPermission extends UserPermission        //定义管理员权限类
{
    public function add ()                          //重写添加方法
    {
        return true;
    }
    public function edit ()                         //重写编辑方法
    {
      return true;
    }
    public function delete ()                       //重写删除方法
```






```
        {
            return true;
        }
    }
class UserPermissionFactory                              //用户权限工厂
{
    public static function getUser ($userType)           //获得用户权限方法
    {
        switch ($userType) {
            case 'GUEST':
                return new GuestPermission();            //返回普通浏览者权限对象
                break;
            case 'MEMBER':
                return new MemberPermission();           //返回会员权限对象
                break;
            case 'ADMIN':
                return new AdminPermission();            //返回管理员权限对象
                break;
        }
    }
}
```

（2）定义二维数组$users，使用该数组保存用户名称及权限类别，代码如下：

```
$users = array(
    0=>array('name'=>'大麦', 'type' => 'GUEST'),
    1=>array('name'=>'SOFT', 'type' => 'MEMBER'),
    2=>array('name'=>'春泉', 'type' => 'GUEST'),
    3=>array('name'=>'金星', 'type' => 'GUEST'),
    4=>array('name'=>'360', 'type' => 'MEMBER'),
    5=>array('name'=>'MR', 'type' => 'ADMIN')
);
```

（3）使用 require 语句包含 UserPermissionFactory.php 及 db.php 文件，然后使用 foreach 循环语句遍历用户数组，并根据用户类别通过用户权限工厂生成相应的用户权限对象，最后打印用户权限信息。实现该过程的代码如下：

```
<?php
require 'UserPermissionFactory.php';                 //包含用户权限工厂所在文件
require 'db.php';                                    //包含用户信息数组所在文件
foreach ($users as $user){                           //遍历用户权限数组
    $userPermission = UserPermissionFactory::getUser($user['type']);     //获得用户权限对象
?>
<!--  以下代码用于打印用户权限信息  -->
<tr>
    <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=$user['name']?></td>
    <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=str_replace('ADMIN', '管理员', str_replace('MEMBER', '会员', str_replace('GUEST', '访客', $user['type'])))?></td>
    <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=$userPermission->select()?'是':'否'?></td>
```





```
        <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=$userPermission->add()?'是
':'否'?></td>
        <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=$userPermission->edit()?'是
':'否'?></td>
        <td style="width:80px; height:22px; border:1px solid #E0056F;"><?=$userPermission->delete()?'
是':'否'?></td>
    </tr>
    <?php
    }
    ?>
```

技术要点

工厂模式是指创建一个类似于工厂的类，通过对类中成员方法的调用返回不同类型的对象。使用工厂模式时，首先创建一个基类，然后根据对象类型的不同来创建不同的扩展类，最后定义工厂类，根据不同的条件来返回相应的对象。工厂模式的实例代码如下：

```
abstract class Father                                   //定义基类
{

}
class Child1 extends Father                             //定义子类 1
{

}
class Child2 extends Father                             //定义子类 2
{

}
class Factory                                           //定义工厂类
{
    public static function create ($condition)          //创建对象
    {
        if ($condition == '条件 1') {
            return new Child1();
        } elseif ($condition == '条件 2') {
            return new Child2();
        }
    }
}
```

上述示例代码中，首先定义抽象基类 Father，然后定义子类 Child1 和 Child2，使之分别继承基类 Father，在实际应用中还应该重写或定义子类 Child1 和 Child2 中的方法，最后定义工厂类 Factory，并在该类中定义静态的 create()方法来根据不同的条件返回相应的对象。

指点迷津：

抽象基类也可以使用接口来代替。



PDO 数据库抽象层

本章读者可以学到如下实例：

- ⏭ 实例 231　连接 MySQL 数据库
- ⏭ 实例 232　连接 MS SQL Server 数据库
- ⏭ 实例 233　连接 Oracle 数据库
- ⏭ 实例 234　通过 PDO 向数据库中添加数据
- ⏭ 实例 235　通过 PDO 浏览数据库中的数据
- ⏭ 实例 236　通过 PDO 更新数据库中的数据
- ⏭ 实例 237　浏览客户留言
- ⏭ 实例 238　明日书店会员注册
- ⏭ 实例 239　添加留言信息
- ⏭ 实例 240　查询留言





实例231　连接MySQL数据库

（实例位置：配套资源\SL\13\231）

实例说明

在学习通过PDO连接MySQL数据库之前，大家先在phpMyAdmin下创建一个MySQL数据库db_database13，并且在db_database13数据库中创建数据表tb_pdo，然后定义数据库连接的参数，最后，通过PDO构造函数创建连接。运行效果如图13.1所示。

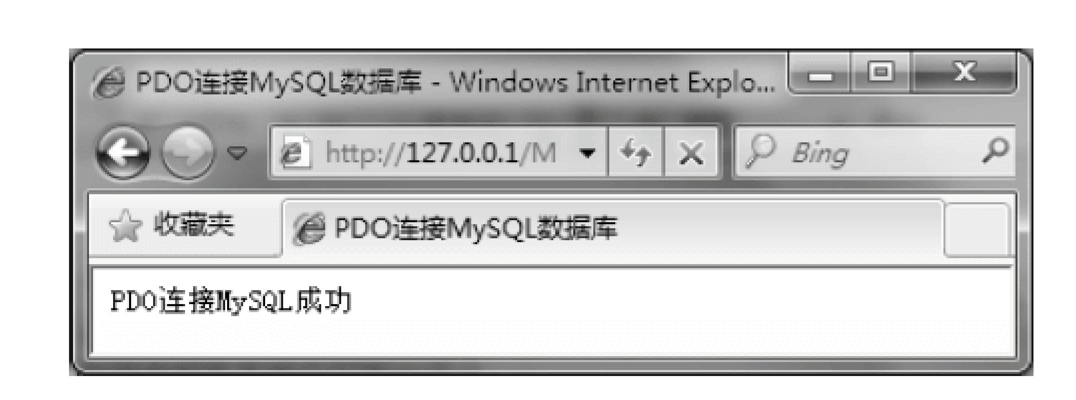


图13.1　连接MySQL数据库

实现过程

具体步骤如下：

（1）创建脚本文件index.php。编写连接MySQL数据库的连接参数，其中包括$dbms、$dbName、$user、$pwd和$host。

（2）实例化PDO对象，输出连接成功的提示并利用try…catch…语句捕获异常。其代码如下：

```
<?php
 header("Content-Type:text/html;charset=utf-8");          //设置页面的编码格式
 $dbms='mysql';                                           //数据库类型
 $dbName='db_database13';                                 //使用的数据库名称
 $user='root';                                            //使用的数据库用户名
 $pwd='111';                                              //使用的数据库密码
 $host='localhost';                                       //使用的主机名称
 $dsn="$dbms:host=$host;dbname=$dbName";
 try {                                                    //捕获异常
        $pdo=new PDO($dsn,$user,$pwd);                    //实例化对象
        echo "PDO连接MySQL成功";
 } catch (Exception $e) {
        echo $e->getMessage()."<br>";
 }
?>
```

技术要点

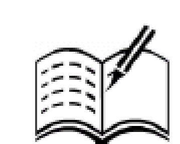
在PDO中，要建立与数据库的连接需要实例化PDO的构造函数，PDO构造函数的语法如下：

```
__construct(string $dsn[,string $username[,string $password[,array $driver_options]]])
```

参数说明：

☑ dsn：数据源名，包括主机名端口号和数据库名称。





Note

☑ username：连接数据库的用户名。

☑ password：连接数据库的密码。

☑ driver_options：连接数据库的其他选项。

实例 232　连接 MS SQL Server 数据库

（实例位置：配套资源\SL\13\232）

实例说明

PDO 这种抽象层的概念可以实现与主流的数据库使用同一接口进行连接。连接的方法是定义指定的 PDO 参数，然后利用同一方法操作即可。本实例通过 PDO 连接 MS SQL Server 数据库，其运行结果如图 13.2 所示。

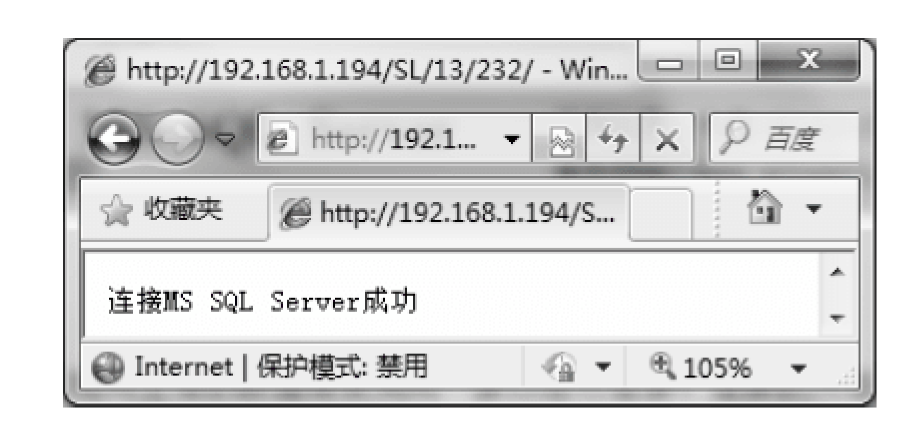


图 13.2　连接 MS SQL Server 数据库

实现过程

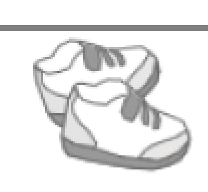
本实例应用 PDO 连接 MS SQL Server 数据库，其操作非常简单，只需更改 PDO 参数 $dbms 为 mssql，$user 为 sa，$pwd 为“”（空）即可。然后使用与连接 MySQL 数据库相同的方法实现连接 SQL Server 数据库。其代码如下：

```
<?php
  header("Content-Type:text/html;charset=utf-8");          //设置页面的编码风格
  $host='PC-201006101638';                                 //设置主机名称
  $user='sa';                                              //设置用户名
  $pwd='';                                                 //设置密码
  $dbName='db_database13';                             //设置所要连接的数据库
  $dbms='mssql';
  $dsn="mssql:host=$host;dbname=$dbName";
  try {                                                //利用 try…catch 捕捉异常并实现连接
        $pdo = new PDO($dsn,$user,$pwd);
        echo "连接 SQL SERVER 成功";
  } catch (Exception $e) {
        die("ERROR!".$e->getMessage()."<br>");
  }
?>
```

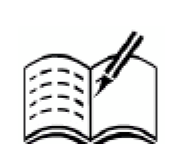
技术要点

（1）如何测试你的 PHP 是否支持 MS SQL Server 数据库，编写一个 phpinfo.php 文件，文件内容如：<?php echo phpinfo(); ?>,运行该文件，即可查看出 PHP 是否支持 MS SQL Server 数据库。

（2）在 php.ini 文件中有关 mssql 的配置已经修改完成后，PHP 仍不支持 mssql 的解决方案：将 PHP 安装目录下的 ntwdblib.dll 文件复制到本机系统盘的 WINDOWS\system32



文件夹下，然后重新启动 Apache 服务器。



Note

实例 233　连接 Oracle 数据库

（实例位置：配套资源\SL\13\233）

实例说明

这里我们连接 Oracle 数据库服务器，数据库名称是 192.168.1.59:1521/oralcles，数据库服务器的用户名是 system，密码为 mrsoft。其运行结果如图 13.3 所示。

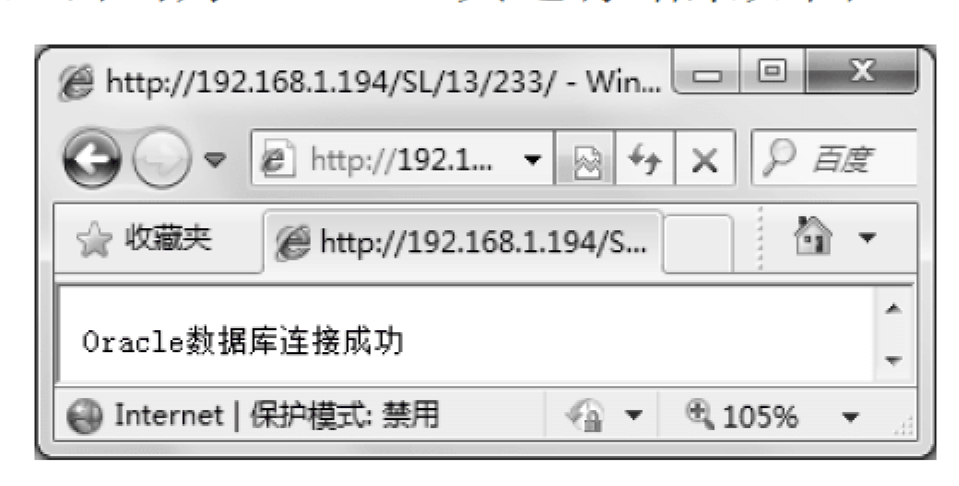


图 13.3　连接 Oracle 数据库

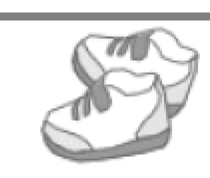
实现过程

首先，定义数据库连接的参数。然后，实例化 PDO 构造函数。最后，通过 try catch 语句判断数据库是否连接成功。其代码如下：

```
<?php
$dbms='oci'; //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$dbName='192.168.1.59:1521/oralcles';          //使用的数据库
$user='system';                                //数据库连接用户名
$pass='mrsoft';                                //对应的密码
$dsn="$dbms:dbname=$dbName";
try {
    $pdo = new PDO($dsn, $user, $pass); //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
    echo "连接成功<br/>";
    $pdo = null;
} catch (PDOException $e) {
    die ("Error!: " . $e->getMessage() . "<br/>");
}
?>
```

技术要点

通过 PDO 连接 Oracle 数据库，首先要在本机上安装 Oracle 数据库的客户端。然后，去掉 php.ini 文件中 extension=php_pdo.dll 和 extension=php_pdo_oci.dll 前面的分号，加载 PDO 模块。最后，将 Oracle 客户端的 oraociei10.dll、oci.dll 和 orannzsbb10.dll 复制到 Apache 的 bin 文件夹下。



实例 234　通过 PDO 向数据库中添加数据

（实例位置：配套资源\SL\13\234）

实例说明

PDO 数据库抽象层的主要特点是为不同的数据库提供统一的接口，使用户在后期维护或数据库变更时减少麻烦。本实例通过 PDO 向已经创建好的数据库中添加数据，其运行效果如图 13.4 所示。

PDO插入数据

用户名：

密　码：

确定　重置　插入数据成功，影响条数为1

图 13.4　通过 PDO 向数据库中添加数据

实现过程

具体步骤如下：

（1）创建脚本文件 index.php，在此脚本文件中编写表单，其中包括文本框、密码框、提交按钮和重置按钮。

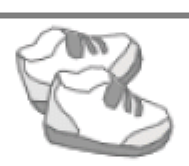
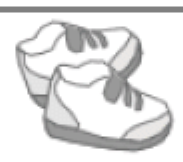
（2）当单击“确定”按钮时，利用$_POST 全局数组获取文本框和密码框的信息，并将其做为 SQL 语句的参数。SQL 语句代码如下：

```
$sql="insert into tb_pdo values('','$_POST[text]','$_POST[pwd]',now())";
```

（3）定义 PDO 抽象层的参数，其中包括$dbms、$user、$pwd、$host 等，拼接$dsn。

（4）实例化 PDO 对象，利用对象句柄调用 exec()函数向数据库中插入数据信息，并显示插入数据的条数。其核心代码如下：

```
<?php
  header("Content-Type:text/html;charset=utf-8");        //设置编码格式
  $dbms='mysql';                                         //定义数据库类型
  $dbName='db_database13';                               //设置连接数据库
  $user='root';                                          //数据库用户名
  $pwd='111';                                            //数据库密码
  $host='localhost';                                     //服务器
  $dsn="$dbms:host=$host;dbname=$dbName";
?>
<?php
if(isset($_POST['sub'])){                                //判断是否提交
  if($_POST['text']==""&&$_POST['pwd']==""){             //判断用户名和密码是否为空
        echo "文本框内容不能为空";
  }else{
        try {
              $pdo=new PDO($dsn,$user,$pwd);             //实例化对象
              $sql="insert into tb_pdo (username,userpwd,date) values('".$_POST['text']."','".$_POST
['pwd']."','".date("Y-m-d")."')";                        //定义添加语句
              $result=$pdo->exec($sql);                  //执行添加语句
              echo "插入数据成功，影响条数为".$result;
        } catch (Exception $e) {
```

```
            echo "ERROR!!".$e->getMessage()."<br>";  //返回错误信息
        }
    }
}
?>
```



Note

技术要点

本实例通过 PDO 中的 exec()方法向数据库中添加数据。exec()方法返回执行后受影响的行数，其语法如下：

```
int PDO::exec ( string statement )
```

参数 statement 是要执行的 SQL 语句。该方法返回执行查询时受影响的行数，通常用于 INSERT、DELETE 和 UPDATE 语句中。

实例 235　通过 PDO 浏览数据库中的数据

（实例位置：配套资源\SL\13\235）

实例说明

利用 PDO 抽象层显示数据需要使用函数 query()，此函数与普通的操作数据库函数并无区别，只不过在 PDO 中此函数只用于数据的查询，而插入、修改和更新的方法需要使用 exec()函数来实现。本实例的运行效果如图 13.5 所示。

ID	用户名称	用户密码	操作时间
1	mr	mrsoft	2010-11-24
2	mrsoft	mrkj	2010-11-24
3	ym	066066	2010-11-24
4	pkh	123456	2010-11-24

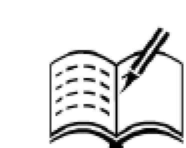
图 13.5　通过 PDO 浏览数据库中的数据

实现过程

首先，创建脚本文件 index.php，编写表格，设置页面的编码。其次，定义 PDO 所需的相关参数，拼接 dsn 参数，利用对象句柄$pdo 调用 PDO 抽象层的 query()方法并返回结果集。最后，将结果集通过 foreach()语句循环输出到表格中。其代码如下：

```
<?php
  header("Content-Type:text/html;charset=utf-8");          //定义页面编码风格
  $dbms='mysql';                                           //定义 PDO 的相关参数
  $dbName='db_database13';
  $user='root';
  $pwd='111';
  $host='localhost';
  $dsn="$dbms:host=$host;dbname=$dbName";
?>
<table>                                                    //表格输出数据内容
<tr>
        <td class="one">ID</td>
        <td class="one">用户名称</td>
```






```
            <td class="one">用户密码</td>
            <td class="one">操作时间</td>
    </tr>
    <?php
      try {                                                   //利用 try…catch…语句捕获异常
            $pdo=new PDO($dsn,$user,$pwd);
            $sql="select * from tb_pdo";
            $result=$pdo->query($sql);
            foreach($result as $key=>$value){
    ?>
      <tr>
            <td><?php echo $value[0];?></td>                  //循环输出数据并定义到表格中
            <td><?php echo $value[1];?></td>
            <td><?php echo $value[2];?></td>
            <td><?php echo $value[3];?></td>
      </tr>
    <?php
            }
      } catch (Exception $e) {
            echo "ERROR!!".$e->getMessage()."<br>";
      }
    ?>
```

技术要点

query()方法通常用于返回执行查询后的结果集。其语法如下：

```
PDOStatement PDO::query ( string statement )
```

参数 statement 是要执行的 SQL 语句，它返回的是一个 PDOStatement 对象。

实例 236　通过 PDO 更新数据库中的数据

（实例位置：配套资源\SL\13\236）

实例说明

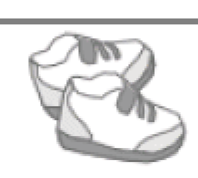
在利用 MySQL 数据库函数更新数据时，需要将要更改的数据信息 id 拼接到地址栏的参数中。通过 PDO 更新数据库中的数据也不例外。在本实例中，单击“修改”超链接，将指定的数据传递到表单中，在表单中对数据进行更新，最后单击“确定”按钮，完成更新操作，其运行效果如图 13.6 和图 13.7 所示。

ID	用户名称	用户密码	操作时间	修改
1	mr	mrsoft	2010-11-24	修改
2	mrsoft	mrkj	2010-11-24	修改
3	ym	066066	2010-11-24	修改
4	pkh	123456	2010-11-24	修改
3	yangming	••••••	2010-11-24	确定

图 13.6　通过 PDO 更新数据库中的数据



图 13.7　更新数据成功





Note

实现过程

首先，创建脚本文件 index.php。编写表格，设置 PDO 抽象层的相关参数拼接$dsn 变量。通过 PDO 中的预处理语句 prepare()和 execute()执行 SQL 查询语句，并且应用 while 语句和 fetch()方法完成数据的循环输出。其关键代码如下：

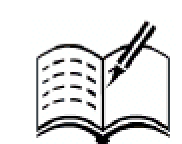
```
<?php
 try {
        $pdo=new PDO($dsn,$user,$pwd);              //实例化对象
        $sql="select * from tb_pdo";                //查询 SQL 语句
        $result=$pdo->prepare($sql);                //准备查询语句
        $result->execute();                         //执行查询语句，并返回结果集
        while($res=$result->fetch(PDO::FETCH_ASSOC)){//循环输出查询结果集，设置结果集为
关联索引
?>
 <tr>
        <td class="one"><?php echo $res['id'];?></td>
        <td class="one"><?php echo $res['username'];?></td>
        <td class="one"><?php echo $res['userpwd'];?></td>
        <td class="one"><?php echo $res['date'];?></td>
        <td class="one"><a href="index.php?id=<?php echo $res['id'];?>">修改</a></td>
 </tr>
```

其次，判断“修改”按钮是否被单击，当“修改”按钮被单击时，获取地址栏传递的参数，根据参数查询数据库中的数据，在表格最下方动态创建<form>表单，包括三个文本框和一个提交按钮，将查询到的数据作为表单的元素值，并设置 id 的文本框为只读。

```
<form action="method="post">
<?php
        }
        if(isset($_GET['id'])){
            $sqls="select * from tb_pdo where id='".$_GET['id']."'";
            $result=$pdo->prepare($sqls);              //准备查询语句
            $result->execute();                        //执行查询语句，并返回结果集
            while($sel=$result->fetch(PDO::FETCH_ASSOC)){//循环输出查询结果集，设置结果
集为关联索引


?>
 <tr>
            <td class="one"><input readonly type='text' name='id' value='<?php echo $sel['id'];?>'></td>
            <td class="one"><input type='text' name='user' value='<?php echo $sel['username'];?>'></td>
            <td class="one"><input type='password' name='pwd' value='<?php echo $sel['userpwd'];?>'>
</td>
            <td class="one"><input type='text' name='date' value='<?php echo $sel['date'];?>'></td>
            <td class="one"><input class="two" type='submit' name='sub1' value='确定'></td>
```






```
    </tr>

<?php
            }
        }
?>
    </form>
```

最后，当单击“确定”按钮时，获取表单提交的数据，通过 PDO 的 exec()方法执行更新操作。其代码如下：

```
<?php
        if(isset($_POST['sub1'])){                          //判断更新按钮是否设置
            $sqles="update tb_pdo set username='".$_POST['user']."', userpwd='".$_POST['pwd']."',
date='".$_POST['date']."' where id='".$_POST['id']."'";     //定义更新语句
            $resu=$pdo->exec($sqles);                       //执行更新语句
            if($resu==1){
                echo "<script>alert('更新数据成功');window.location.href='index.php'</script>";
            }
        }
    } catch (Exception $e) {
        echo "ERROR!!!".$e->getMessage()."<br>";          //返回错误信息
    }
?>
```

技术要点

预处理语句包括 prepare()和 execute()两个方法。首先，通过 prepare()方法做查询的准备工作，然后，通过 execute()方法执行查询。并且还可以通过 bindParam()方法来绑定参数提供给 execute()方法。其语法如下：

```
PDOStatement PDO::prepare ( string statement [, array driver_options] )
bool PDOStatement::execute ( [array input_parameters] )
```

多学两招：

预处理语句，它是要运行的 SQL 的一种编译过的模板，它可以使用变量参数进行定制。预处理语句可以带来两大好处：

查询只需解析（或准备）一次，但是可以用相同或不同的参数执行多次。当查询准备好后，数据库将分析、编译和优化执行该查询的计划。对于复杂的查询，这个过程要花比较长的时间，如果需要以不同参数多次重复相同的查询，那么该过程将大大降低应用程序的速度。通过使用预处理语句，可以避免重复分析/编译/优化周期。简言之，预处理语句使用资源更少，因而运行得更快。

提供给预处理语句的参数不需要用引号括起来，驱动程序会处理这些。如果应用程序独占地使用预处理语句，那么可以确保没有 SQL 入侵发生。但是，如果你仍然将查询的其他部分建立在不受信任的输入之上，那么就仍然存在风险。



指点迷津：

PDO 中执行 SQL 语句方法的选择。

（1）如果只是执行一次查询，那么 PDO->query 是较好的选择。虽然它无法自动转义发送给它的任何数据，但是它在遍历 SELECT 语句的结果集方面是非常方便的。然而在使用这个方法时也要相当小心，因为如果没有在结果集中获取到所有数据，那么下次调用 PDO->query 时可能会失败。

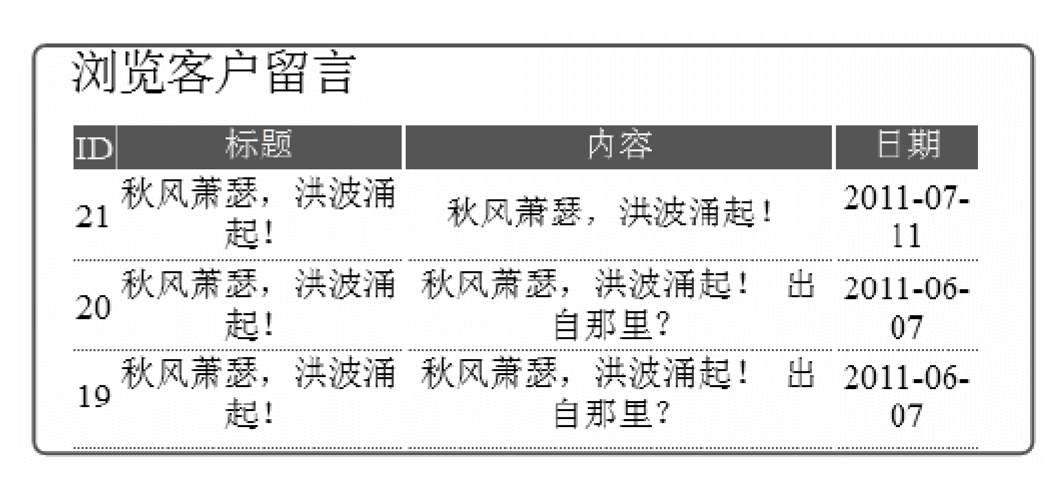
（2）如果多次执行 SQL 语句，那么最理想的方法是 prepare()和 execute()。这两个方法可以对提供给它们的参数进行自动转义，进而防止 SQL 注入攻击；同时由于在多次执行 SQL 语句时，应用的是预编译语句，还可以减少资源的占用，提高运行速度。

实例 237　浏览客户留言

（实例位置：配套资源\SL\13\237　视频位置：配套资源\SP\13\237）

实例说明

本实例通过 PDO 中的 fecth()方法获取结果集中下一行的数据，进而应用 while 语句完成数据库中客户留言信息的循环输出，同时在定义 SQL 语句时对数据按 ID 降幂排列，显示 4 条记录。其运行效果如图 13.8 所示。


浏览客户留言

ID	标题	内容	日期
21	秋风萧瑟，洪波涌起！	秋风萧瑟，洪波涌起！	2011-07-11
20	秋风萧瑟，洪波涌起！	秋风萧瑟，洪波涌起！出自那里？	2011-06-07
19	秋风萧瑟，洪波涌起！	秋风萧瑟，洪波涌起！出自那里？	2011-06-07

图 13.8　浏览客户留言

实现过程

具体步骤如下：

（1）创建脚本文件 index.php。定义 PDO 抽象层的相关参数。其中包括$host、$user、$pwd、$dbName、$dbms 等。

（2）在 try…catch 语句内部实例化 PDO 对象，通过 PDO 中的预处理语句 prepare()和 execute()执行 SQL 查询语句，并且应用 while 语句和 fetch()方法完成数据的循环输出。其代码如下：

```
<?php
  $dbms="mysql";
  $user="root";
  $pwd="111";
  $host="localhost";
  $dbName="db_database13";
  $dsn="$dbms:host=$host;dbname=$dbName";
  try {
        $pdo = new PDO($dsn,$user,$pwd);
        $pdo->query("SET NAMES utf8");                    //设置编码格式
        $sql="select * from tb_fb order by id desc limit 4";     //定义查询语句
```



```
                $result=$pdo->prepare($sql);                              //准备查询语句
                $result->execute();                                        //执行查询语句，并返回结果集
                while($res=$result->fetch(PDO::FETCH_ASSOC)){//循环输出查询结果集，并且设置结果
集为关联索引
        ?>
            <tr>
                <td class="two"><?php echo $res['id'];?></td>
                <td class="two"><?php echo $res['title'];?></td>
                <td class="two"><?php echo $res['content'];?></td>
                <td class="two"><?php echo $res['date'];?></td>
            </tr>
        <?php
                }
            } catch (Exception $e) {
                echo "ERROR!!!".$e->getMessage()."<br>";
            }
        ?>
```

技术要点

fetch()方法获取结果集中的下一行，其语法格式如下：

```
mixed PDOStatement::fetch ( [int fetch_style [, int cursor_orientation [, int cursor_offset]]] )
```

参数说明：

☑　fetch_style：控制结果集的返回方式，其可选方式如表 13.1 所示。

表 13.1　fetch_style 控制结果集的可选值

值	说　明
PDO::FETCH_ASSOC	关联数组形式
PDO::FETCH_NUM	数字索引数组形式
PDO::FETCH_BOTH	两者数组形式都有，这是默认的
PDO::FETCH_OBJ	按照对象的形式，类似于以前的 mysql_fetch_object()
PDO::FETCH_BOUND	以布尔值的形式返回结果，同时将获取的列值赋给 bindParam()方法中指定的变量
PDO::FETCH_LAZY	以关联数组、数字索引数组和对象 3 种形式返回结果

☑　cursor_orientation：PDOStatement 对象的一个滚动游标，可用于获取指定的一行。

☑　cursor_offset：游标的偏移量。

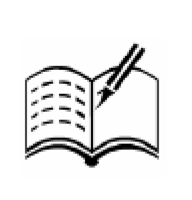
实例 238　明日书店会员注册

（实例位置：配套资源\SL\13\238　视频位置：配套资源\SP\13\238）

实例说明

本实例利用 PDO 抽象层知识实现明日书店会员注册系统。其核心思想是拼接插入的





SQL 代码，将文本框的相关信息动态地添加到数据表中。运行效果如图 13.9 所示。



图 13.9　明日书店会员注册

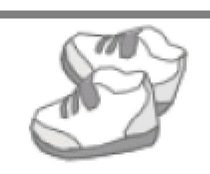
实现过程

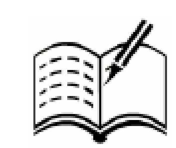
首先，创建脚本文件 index.php。在脚本文件中编写表单，设置一个提交按钮和一个重置按钮。其次，当单击“注册”按钮时，首先通过 PDO 连接 MySQL 数据库，实例化 PDO 对象并利用对象句柄调用 exec()方法，向数据库中添加数据。其代码如下：

```
<?php
 $dbms='mysql';
 $host='localhost';
 $user='root';
 $pwd='111';
 $dbName='db_database13';
 $dsn="$dbms:host=$host;dbname=$dbName";
 if(isset($_POST['sub'])){                                //判断页面是否存在 sub 变量
     try {
         $pdo=new PDO($dsn,$user,$pwd);                   //实例化 PDO 类
         $sql="insert into tb_zc(username,userpwd,qq,email,date)values('".$_POST['text']."','".$_POST
['pwd']."', '".$_POST['qq']."','".$_POST['mail']."','".date("Y-m-d")."')";     //定义添加语句
         $result=$pdo->exec($sql);                        //执行添加语句
         if($result==1){
             echo "<script>alert('注册成功');</script>";
         }
     } catch (Exception $e) {
         echo "ERROR".$e->getMessage()."<br>";
     }
 }
?>
```

技术要点

本实例通过实例化 PDO 对象并应用 exec()方法执行数据库插入操作，根据返回的结果集$result 来判断用户是否注册成功，如果返回的结果集为“1”，则说明用户注册成功；否则说明用户注册失败。





实例 239　添加留言信息

（实例位置：配套资源\SL\13\239　视频位置：配套资源\SP\13\239）

实例说明

本实例应用 PDO 抽象层操作 MySQL 数据库，实现留言信息的添加功能。当输入标题和内容后，单击“发布”按钮，留言发布成功。其运行效果如图 13.10 所示。

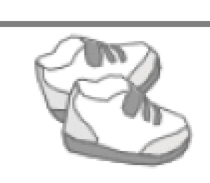


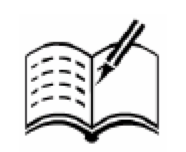
图 13.10　添加留言信息

实现过程

首先，创建脚本文件 index.php。在脚本文件中编写表单，添加标题的文本框、内容的文本域以及“发布”按钮。其次，当单击“发布”按钮时，首先通过 PDO 连接 MySQL 数据库，实例化 PDO 对象并利用对象句柄调用 exec()方法，向数据库中添加数据。其代码如下：

```
<?php
  $dbms='mysql';                    //设置数据库类型
  $user='root';                     //定义数据库用户
  $pwd='111';                       //定义数据库密码
  $host='localhost';                //定义服务器
  $dbName='db_database13';          //定义连接数据库
  $dsn="$dbms:host=$host;dbname=$dbName";
  if(isset($_POST['sub'])){         //判断发布按钮是否被设置
        try {
              $pdo = new PDO($dsn,$user,$pwd);                //实例化 PDO 类
              $pdo->query("SET NAMES utf8");                  //设置数据库编码
              $sql="insert  into  tb_fb(title,content,date)values('".$_POST['title']."',  '".$_POST['content']."','
".date("Y-m-d")."')";
              $rs=$pdo->exec($sql);                           //执行添加语句
              if($rs=="1"){
                    echo "<b>新闻发布成功</b>";
              }
        } catch (Exception $e) {
              echo "ERROR".$e->getMessage()."<br>";
        }
  }
?>
```





技术要点

本实例应用 query()方法执行 SET NAMES utf8 语句，设置页面的编码格式，以避免用户在执行添加数据操作过程中，向数据库插入乱码；同时也避免了应用 PDO 读取数据时，在页面中显示乱码的问题。

实例 240　查询留言

（实例位置：配套资源\SL\13\240　视频位置：配套资源\SP\13\240）

实例说明

本实例实现了一个站内搜索的功能，对站内的留言信息进行模糊查询。通过 PDO 中的 fecthAll()方法获取结果集中所有行，并且通过 for 语句读取二维数组中的数据，完成查询结果的循环输出。其运行效果如图 13.11 所示。

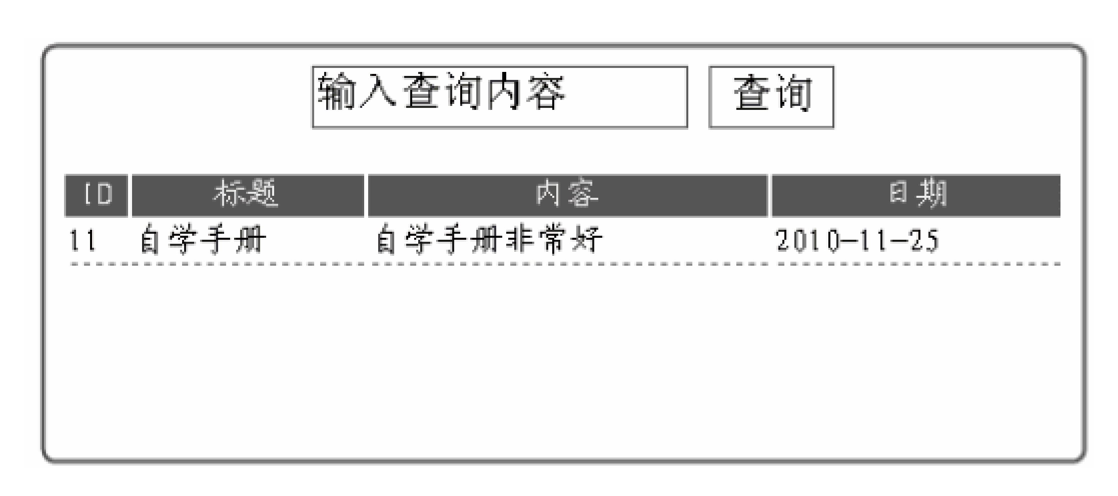


图 13.11　查询留言

实现过程

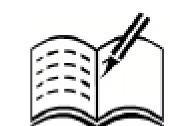
具体步骤如下：

（1）创建脚本文件 index.php。定义表单，提交查询的关键字。

（2）当单击“查询”按钮时，首先，判断查询关键字是否为空；其次，使用 PDO 抽象层连接 MySQL 数据库，并在 try…catch 内部利用 PDO 对象句柄调用预处理语句 prepare()和 execute()执行模糊查询；最后应用 fecthAll()方法获取结果集中所有行，通过 for 语句循环输出查询结果。其核心代码如下：

```
<?php
 if(isset($_POST['sub'])){                                              //判断查询按钮是否被设置
        if(!isset($_POST['text']) || $_POST['text']=="输入查询内容"){ //判断查询关键字是否为真
              echo "文本框内容不能为空";
        }else{
              $dbms="mysql";
              $user="root";
              $pwd="111";
              $host="localhost";
              $dbName="db_database13";
              $dsn="$dbms:host=$host;dbname=$dbName";
              try {
```






```
                $pdo = new PDO($dsn,$user,$pwd);                //实例化 PDO 类
                $sql="select * from tb_fb where title like '%".$_POST['text']."%'";//定义查询语句
                $pdo->query("SET NAMES utf8");                  //设置编码格式
                $result=$pdo->prepare($sql);                    //准备查询语句
                $result->execute();                             //执行查询语句，并返回结果集
                $res=$result->fetchAll(PDO::FETCH_ASSOC);       //获取结果集中的所有数据
                for($i=0;$i<count($res);$i++){                  //循环读取二维数组中的数据
?>
        <tr>
            <td class="f"><?php echo $res[$i]['id'];?></td>
            <td class="f"><?php echo $res[$i]['title'];?></td>
            <td class="f"><?php echo $res[$i]['content'];?></td>
            <td class="f"><?php echo $res[$i]['date'];?></td>
        </tr>
<?php
                }
            } catch (Exception $e) {
                echo "ERROR".$e->getMessage()."<br>";
            }
        }
    }
?>
```

技术要点

fetchAll()方法获取结果集中的所有行。其语法如下：

```
array PDOStatement::fetchAll ( [int fetch_style [, int column_index]] )
```

参数说明：

☑ fetch_style：控制结果集中数据的显示方式。

☑ column_index：字段的索引。

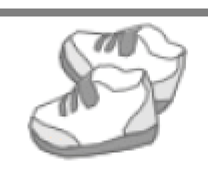
其返回值是一个包含结果集中所有数据的二维数组。



Smarty 模板

本章读者可以学到如下实例：

- ⏭ 实例 241　封装 Smarty 模板的配置方法
- ⏭ 实例 242　Smarty 模板中日期、时间的格式化输出
- ⏭ 实例 243　Smarty 模板中的编码
- ⏭ 实例 244　Smarty 模板制作日期、时间选择器
- ⏭ 实例 245　通过 Section 循环输出数据
- ⏭ 实例 246　开启网站注册页面的缓存
- ⏭ 实例 247　通过配置文件定义变量
- ⏭ 实例 248　Smarty+ADODB 完成数据的分页显示
- ⏭ 实例 249　Smarty 模板中 truncate()方法截取字符串
- ⏭ 实例 250　Register_function()方法注册模板函数
- ⏭ 实例 251　Smarty 模板中的关键字描红技术
- ⏭ 实例 252　Smarty 模板中生成数字验证码



实例 241　封装 Smarty 模板的配置方法

（实例位置：配套资源\SL\14\241　视频位置：配套资源\SP\14\241）

实例说明

本实例中向读者介绍一种 Smarty 模板的配置方法，使用这种方法无论实例的文件夹怎么改变，都不会影响到实例的运行。其运行效果如图 14.1 所示。



图 14.1　应用封装 Smarty 模块的配置方法运行程序的效果图

实现过程

具体步骤如下：

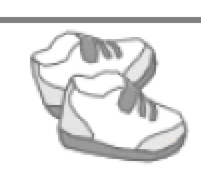
（1）在实例的根目录下创建 system 文件夹，存储 Smarty 的配置文件 system.smarty.inc.php 和实例化操作文件 system.inc.php，同时创建 Smarty 文件夹，在 Smarty 文件夹下创建 templates_c、configs 和 cache 3 个目录。

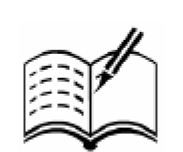
（2）创建 index.php 文件，重新载入 Smarty 的配置文件，即通过 include 语句包含 system.inc.php。在 PHP 脚本中读取文本文件中的数据，并将其赋给模板变量，指定模板页。其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
include("system/system.inc.php");                                 //包含配置文件
/*   使用 Smarty 赋值方法将一对名称/方法发送到模板中   */
$smarty->assign('title','第一个 Smarty 程序');
$str = file_get_contents('files/content.txt');                    //读取文本文件中的数据
$smarty->assign('content',iconv("gb2312","utf-8",$str));//将文本文件中的数据存储到模板变量中
/*   显示模板   */
$smarty->display('index.html');
?>
```

（3）编写 index.html 模板文件，输出模板变量传递的数据。其他的代码请参考配套资源中的源程序，这里不再赘述。

为了让读者更好地理解 Smarty 模板的配置方法和文件的存储位置，下面给出本实例





Note

的文件夹架构，如图 14.2 所示。

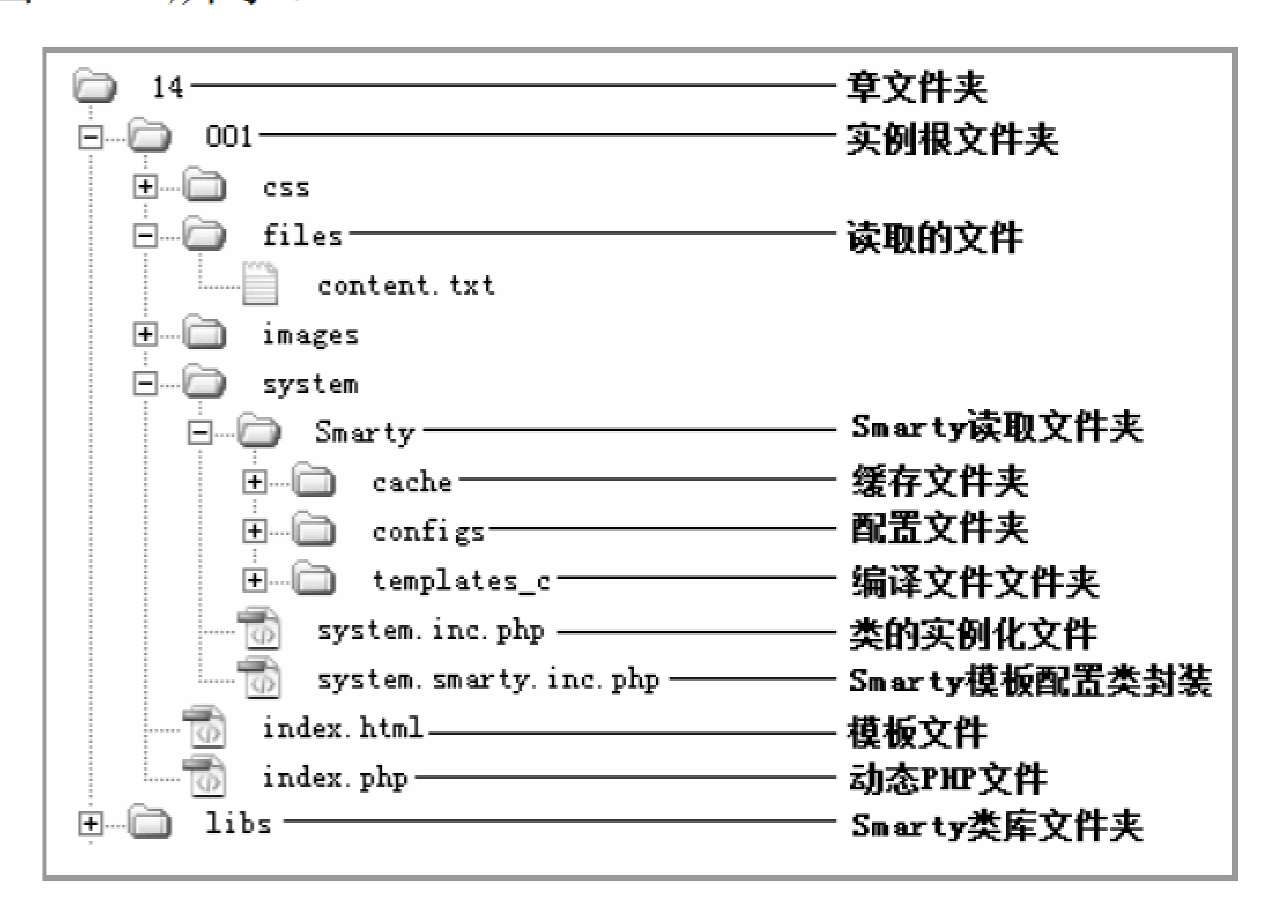


图 14.2　实例 241 的文件夹架构

技术要点

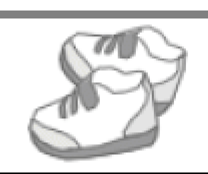
这里将 Smarty 配置方法封装到类中，并存储到 system.smarty.inc.php 文件中。首先，包含 Smarty 类文件 Smarty.class.php；然后，定义 SmartyProject 类，继承 Smarty 父类；最后，定义 SmartyProject()方法，设置 Smarty 中模板文件（templates）、编译文件（templates_c）、配置文件（configs）和缓存文件（cache）的存储位置。这就是新的配置文件 system.smarty.inc.php，其代码如下：

```
<?php
require("../libs/Smarty.class.php");          //调用 Smarty 文件
class SmartyProject extends   Smarty{          //定义类，继承 Smarty 父类
  function SmartyProject(){                    //定义方法，配置 Smarty 模板
        $this->template_dir = "./";           //指定模板文件存储在根目录下
        $this->compile_dir = "./system/Smarty/templates_c/";     //指定编译文件存储位置
        $this->config_dir = "./system/Smarty/configs/";
        $this->cache_dir = "./system/Smarty/cache/";
    }
}
?>
```

既然已经将 Smarty 的配置方法存储到一个类中，那么就需要对类进行实例化，根据返回的对象名称调用 Smarty 中的方法，类的实例化操作在 system.inc.php 文件中完成，其返回对象名为$smarty。其代码如下：

```
<?php
  require("system.smarty.inc.php");          //调用类文件
  $smarty=new SmartyProject();               //执行类的实例化操作
?>
```

通过此方法配置 Smarty 模板的好处是：无论将程序复制到哪个服务器下执行，都不需要更改服务器或者 Smarty 文件的绝对路径，程序都可以直接运行。



指点迷津：

这种配置方法中调用的是存储在目录 14 下的 libs 文件夹中的 Smarty 类库，同时在这种方法中将配置文件存储于实例根目录下的 system 文件夹中，将模板文件夹（templates）、编译文件夹（templates_c）、配置文件夹（configs）和缓存文件夹（cache）存储在 system\Smarty 目录下。



实例 242　Smarty 模板中日期、时间的格式化输出

（实例位置：配套资源\SL\14\242）

实例说明

在 PHP 脚本中日期、时间的格式化输出最常用的就是 date()函数，那么在 Smarty 模板中该如何完成日期、时间的输出呢？这就是本实例中要讲解的内容，通过 Smarty 模板中的 date_format()函数完成日期、时间的格式化输出，其运行结果如图 14.3 所示。



图 14.3　Smarty 模板中日期的格式化输出

实现过程

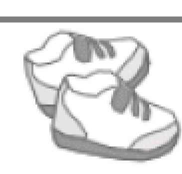
本实例首先创建一个 index.html 模板页面，然后在 index.html 模板文件中插入 date_format()函数，并输出系统的当前时间。其关键代码如下：

```
        <td width="480" height="18" align="left">当前时间：{$smarty.now|date_format:" %A   %B - %e - %Y"}</td>
        <!--{$smarty.now|date_format}
        {$smarty.now|date_format:"%A, %B %e, %Y":$times}
        {$smarty.now|date_format:"%H:%M:%S"}
        {$yesterday|date_format}
        {$yesterday|date_format:"%A, %B %e, %Y"}
        -->
```

本实例的其他内容可以参考配套资源中的源程序，这里不再赘述。

技术要点

（1）Smarty 模板中的 date_format()函数，格式化从函数 strftime()获得的时间和日期。



其应用示例如下：

```
{$smarty.now|date_format:"%A, %B %e, %Y":$times}
```

参数说明：

☑ $smarty.now：传递给 date_format()的数据。这个参数可以是在 PHP 动态页中定义的日期、时间模板变量，也可以使用 Smarty 中的保留变量$smarty.now 或者 Smarty 模板中的日期。

☑ %A, %B %e, %Y：date_format()函数执行格式化操作时使用的格式。

☑ $times：当传递给 date_format()的数据为空时，通过$times 设置 date_format()格式化的默认值。

（2）date_format()函数可以使用的转换格式很多，其常用的转换格式说明如下：

```
%a - 根据当地格式输出“星期”缩写格式
%A - 根据当地格式输出“星期”全称格式
%b - 根据当地格式输出“月”缩写格式
%B - 根据当地格式输出“月”全称格式
%Y - 根据当地格式输出“年”全称格式
```

要了解 date_format()函数所有可以使用的转换格式，建议读者参考 Smarty 模板的参考手册。

实例 243　Smarty 模板中的编码

（实例位置：配套资源\SL\14\243）

实例说明

在 PHP 脚本中应用 urlencode()函数对超链接中传递的参数进行编码，而在 Smarty 模板中提供 escape()函数对字符串进行编码。在本实例就详细地讲解一下这个函数的应用，通过它完成对超级链接中传递的参数进行编码，其运行结果如图 14.4 所示。

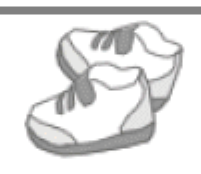


图 14.4　Smarty 模板中对超链接的参数值进行编码

实现过程

具体步骤如下：

（1）这里首先创建一个台管理系统主页的模板，为不同模块之间跳转的超链接传递



的参数值进行编码，其具体的操作在 main.html 模板页中完成，关键代码如下：

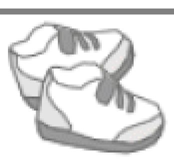
```
        <map name="Map" id="Map">
        <area shape="rect" coords="29,41,88,62" href="main.php?caption={"商品添加"|escape:"url"}&amp;
type={"商品管理"}" />
        <area shape="rect" coords="30,71,91,90" href="main.php?caption={"商品修改"|escape:"url"}&amp;
type={"商品管理"}" />
        <area shape="rect" coords="31,99,91,118" href="main.php?caption={"商品删除"|escape:"url"}&amp;
type={"商品管理"}" />
        </map>
        <map name="Map2" id="Map2">
        <area shape="rect" coords="30,45,97,63" href="main.php?caption={"会员删除"|escape:"url"}&amp;
type={"会员管理"}" />
        <area shape="rect" coords="34,80,89,98" href="main.php?caption={"会员浏览"|escape:"url"}&amp;
type={"会员管理"}" />
        </map>
        <map name="Map5" id="Map5">
        <area shape="rect" coords="12,11,54,36" href="logout.php" />
        </map>
        <map name="Map6" id="Map6">
        <area shape="rect" coords="14,12,51,36" href="main.php?caption={"商品修改"|escape:"url"}&amp;
type={"商品管理"|escape:"url"}" />
        </map>
```



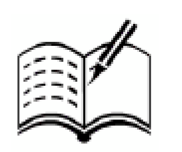
（2）虽然通过 escape()编码后的模板变量输出时是乱码，但是在 switch 语句中仍然可以直接使用$_GET['caption']获取超链接的值，根据这个值作出判断，在主页中输出那个模块的内容。不需要对这个参数值进行解码。

执行判断是在 main.php 文件中完成的，其关键代码如下：

```
        <?php
        session_start();                                    //初始化 SESSION 变量
        if($_SESSION['user']!="" and $_SESSION['pass']!=""){   //判断用户名和密码是否为空
        require("system/system.inc.php");                   //包含 PHP 配置文件
        switch ($_GET['caption']){                          //根据超链接传递的值进行判断
          case "商品添加";
              include "sho_insert.php";                     //包含 PHP 文件
              $smarty->assign('admin_phtml','sho_insert.html');  //将 PHP 文件对应的模板文件的名称赋
给模板变量
          break;                                            //跳出循环
        //省略了部分代码
          default:
              include "sho_update.php";
              $smarty->assign('admin_phtml','sho_update.html');
          break;
        }
        $smarty->assign("title","后台管理系统--".$_GET['caption']);     //为模板变量赋值
        $smarty->assign("caption",$_GET['caption']);        //为模板变量赋值，输出模块名称
        $smarty->assign("type",$_GET['type']);              //为模板变量赋值，输出模块所属类别
        $smarty->assign("user",$_SESSION['user']);          //为模板变量赋值，输出登录用户名称
```



```
    $smarty->display("main.html");                         //指定模板页
    }else{
        echo "<script>alert('您不具备访问权限！'); window.location.href='index.html';</script>";
    }
    ?>
```



Note

技术要点

escape()函数用于 html 转码、url 转码，在没有转码的变量上转换单引号、十六进制转码或者 javascript 转码。默认是 html 转码。其基本的应用格式如下：

```
{$articleTitle|escape}
{$articleTitle|escape:"html"} {* escapes & " ' < > *}
{$articleTitle|escape:"htmlall"} {* escapes ALL html entities *}
{$articleTitle|escape:"url"}
{$articleTitle|escape:"quotes"}
<a href="mailto:{$EmailAddress|escape:"hex"}">{$EmailAddress|escape:"hexentity"}</a>
```

实例 244　Smarty 模板制作日期、时间选择器

（实例位置：配套资源\SL\14\244）

实例说明

在 Smarty 模板中，提供自定义的日期、时间处理函数，通过它们可以得到不同格式的日期、时间信息。在本实例中将应用 Smarty 模板中的日期时间选择函数，设计一个日期时间选择器，记录用户登录的时间。其运行结果如图 14.5 所示。



图 14.5　Smarty 模板制作日期、时间选择器

实现过程

具体步骤如下：

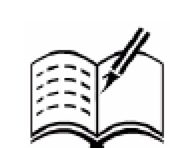
（1）创建 system 文件夹。首先，定义 system.smarty.inc.php 文件，封装 Smarty 的配置方法以及 ADODB 连接和操作数据库的方法。然后，定义 system.inc.php 文件，对 Smarty 配置类、数据库的连接和操作类进行实例化，并返回连接对象。最后，创建 Smarty 文件夹，定义 Smarty 的编译文件、配置文件和缓存文件的存储目录。

（2）创建 index.php 文件。首先，设置页面的编码格式、载入配置文件。然后，生成随机验证码，并且将随机验证码的值赋给模板变量。最后，指定 index.html 模板页。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
```



```
require_once("system/system.inc.php");                      //包含配置文件
$array= explode(' ', mt_rand(1000,9999));                    //生成随机验证码
$smarty->assign('title','Smarty 模板制作日期、时间选择器');   //将指定数据赋给模板变量
$smarty->assign('content',$array);                           //将数组赋给模板变量
$smarty->display('index.html');                              //指定模板页
?>
```

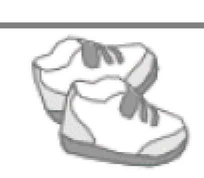


（3）创建 index.html 模板页，设计用户登录的 form 表单。其中，应用 html_select_date() 和 html_select_time()函数生成日期、时间下拉列表框，应用 foreach 语句输出验证码，最终将表单中数据提交到 index_ok.php 文件中，完成用户登录的操作。其关键代码如下：

```
<form id="form" name="form" method="post" action="index_ok.php" onsubmit="return check_form(); ">
    日期：{html_select_date}
    时间：{html_select_time use_24_hours=true}
    用户名：<input name="user" type="text" id="user" size="20" />
    密码：<input name="pass" type="password" id="pass" size="20" />
    验证码：<input type="hidden" id="checks" name="checks" value="{foreach key=key item=item from=$content} {$item} {/foreach}" />
    <input name="check" type="text" id="check" size="8" />{foreach key=key item=item from=$content} {$item} {/foreach}
    <input type="image" name="imageField3" src="images/reg_07.jpg" />
    <input type="image" name="imageField4" onclick="form.reset();return false;" src="images/reg_09.jpg" />
</form>
```

（4）创建 index_ok.php 文件，获取表单中提交的用户登录信息，定义查询语句验证用户提交的用户名和密码是否正确，如果正确则根据日期、时间选择器中提交的数据，更新用户注册信息表中的最后登录时间。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                 //设置文件编码格式
require_once("system/system.inc.php");                               //包含配置文件
if($_POST['user']!=""&&$_POST['pass']!=""&&$_POST['checks']!=""){
    $sql="select * from tb_user where user='".$_POST['user']."' and pass='".md5($_POST['pass'])."'";
    $res=$admindb->ExecSQL($sql,$conn);                              //执行 select 查询语句
    if($res){
        $login_date=$_POST['Date_Year']."-".$_POST['Date_Month']."-".$_POST['Date_Day']." ".$_POST['Time_Hour'].":".$_POST['Time_Minute'].":".$_POST['Time_Second'];
        $sqls="update tb_user set login_date='".$login_date."' where user='".$_POST['user']."' and pass='".md5($_POST['pass'])."'";
        $rs=$admindb->ExecSQL($sqls,$conn);                          //执行 update 更新语句
        echo "<script>alert('用户登录成功！'); window.location.href='main.php';</script>";
    }else{
        echo "<script>alert('用户登录失败！'); window.location.href='index.php';</script>";
    }
}else{
    echo "<script>alert('用户登录信息不可为空！'); window.location.href='index.php';</script>";
}
?>
```



（5）创建 main.php 和 main.html 文件，作为用户登录成功的跳转页面，其代码请参考本书配套资源。



Note

技术要点

在 Smarty 模板中通过自定义函数 html_select_date()和 html_select_time()生成日期、时间选择器。

html_select_date()函数，用于创建日期下拉菜单，可以显示任意年月日。html_select_date()函数的语法说明如表 14.1 所示。

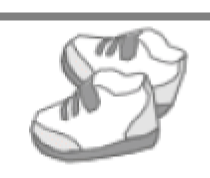
表 14.1　html_select_date()函数的语法说明

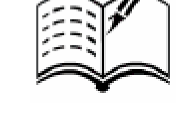
属　　性	类　　型	是 否 必 须	默 认 值	说　　明
year_size	string	No	null	如果设置，为标签添加大小属性
all_extra	string	No	null	如果设置，为所有标签添加附加属性
day_extra	string	No	null	如果设置，为标签添加附加属性
month_extra	string	No	null	如果设置，为标签添加附加属性
year_extra	string	No	null	如果设置，为标签添加附加属性
field_order	string	No	MDY	显示区域的顺序
field_separator	string	No	\n	各区域间输出的分隔字符串
month_value_format	string	No	%m	月份值的 strftime 表示方法，默认为 %m
year_size	string	No	null	如果设置，为标签添加大小属性

html_select_time()函数，创建时间下拉菜单，它可以显示任意时分秒。html_select_time()函数的语法说明如表 14.2 所示。

表 14.2　html_select_time()函数的语法说明

属性	类　　型	是 否 必 须	默 认 值	说　　明
prefix	string	No	Time_	变量名称前缀
time	timestamp	No	UNIX	使用时间类型（data/time）
display_hours	boolean	No	true	是否显示小时
display_minutes	boolean	No	true	是否显示分钟
display_seconds	boolean	No	true	是否显示秒
display_meridian	boolean	No	true	是否显示正午界（上午/下午）
use_24_hours	boolean	No	true	是否使用 24 小时制
minute_interval	integer	No	1	分钟下拉列表的间隔
second_interval	integer	No	1	秒钟下拉列表的间隔
field_array	string	No	n/a	输出值到该值指定的数组
all_extra	string	No	null	如果设置，为标签添加附加属性
hour_extra	string	No	null	如果设置，为标签添加附加属性
minute_extra	string	No	null	如果设置，为标签添加附加属性
second_extra	string	No	null	如果设置，为标签添加附加属性
meridian_extra	string	No	null	如果设置，为标签添加附加属性





实例 245　通过 Section 循环输出数据

（实例位置：配套资源\SL\14\245　视频位置：配套资源\SP\14\245）

实例说明

在 PHP 动态页中，可以通过 while、do…while、for 和 foreach 语句实现数据的循环输出，而在 Smarty 中它也有属于自己的循环输出语句 foreach 和 section。在本实例中将介绍通过 section 语句完成数据的循环输出，运行结果如图 14.6 所示。



图 14.6　通过 section 循环输出数据

实现过程

具体步骤如下：

（1）本实例应用开发的后台管理系统主页为基础，首先去除后台管理系统的登录功能，直接在 index.php 和 index.html 中创建后台管理系统的主页，在 switch 语句中，将 vip_look.php 和 vip_look.html 设置为默认值。index.php 的关键代码如下：

```
<?php
require("system/system.inc.php");
switch ($_GET['caption']){
  case "会员删除";
        include "vip_delete.php";
        $smarty->assign('admin_phtml','vip_delete.html');
  break;
  case "会员浏览";
        include "vip_look.php";
        $smarty->assign('admin_phtml','vip_look.html');
  break;
//省略了部分代码
  default:
        include "vip_look.php";
        $smarty->assign('admin_phtml','vip_look.html');
  break;
}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);               //执行查询操作
if($res){
  $smarty->assign("res",$res);                    //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                  //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);             //执行查询操作
if($res){
  $smarty->assign("res",$res);                  //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                  //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);             //执行查询操作
if($res){
  $smarty->assign("res",$res);                  //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);        //执行查询操作
if($res){
  $smarty->assign("res",$res);                  //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                  //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);             //执行查询操作
if($res){
  $smarty->assign("res",$res);                  //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




Note

```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);               //执行查询操作
if($res){
  $smarty->assign("res",$res);                    //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);           //执行查询操作
if($res){
  $smarty->assign("res",$res);                //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align="left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);              //执行查询操作
if($res){
  $smarty->assign("res",$res);                    //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
     <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
     <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
     <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
     <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




Note

```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);               //执行查询操作
if($res){
  $smarty->assign("res",$res);                    //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```





```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);           //执行查询操作
if($res){
  $smarty->assign("res",$res);                //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                   //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);              //执行查询操作
if($res){
  $smarty->assign("res",$res);                   //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




Note

```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                  //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);             //执行查询操作
if($res){
  $smarty->assign("res",$res);                  //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```




```
$smarty->assign("title","后台管理系统--Section 循环输出数据--".$_GET['caption']);
if($_GET['caption']!=""){
        $smarty->assign("caption",$_GET['caption']);
        $smarty->assign("type",$_GET['type']);
}else{
        $smarty->assign("caption","会员浏览");
        $smarty->assign("type","会员管理");
}
$smarty->assign("dates",date("Y 年 m 月 d 日"));
$smarty->display("index.html");
?>
```

（2）在本实例中对会员浏览模块（vip_look.php 和 vip_look.html）进行实际地开发。即应用 section 语句循环输出数据库中存储的会员信息。

首先，重新创建 vip_look.php 文件，定义 SQL 语句，调用数据库操作类（AdminDB）中的 ExecSQL 方法，查询出数据库中的会员数据，并且将返回的数组赋给指定的模板变量。其代码如下：

```
<?php
$sql="select * from tb_user ";                    //定义 SQL 查询语句
$res=$admindb->ExecSQL($sql,$conn);               //执行查询操作
if($res){
  $smarty->assign("res",$res);                    //将查询结果赋给指定的模板变量
}
?>
```

然后，重新创建 vip_look.html 文件，应用 section 语句循环输出模板变量中传递的会员数据。其关键代码如下：

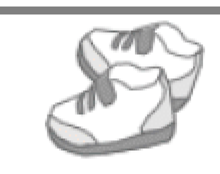
```
{section name=id loop=$res}
  <tr>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].user}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].email}</td>
    <td style="padding-bottom:5px; padding-left:5px; padding-right:5px; padding-top:5px;" align=
"left" bgcolor="#FFFFFF">{$res[id].dates}</td>
  </tr>
{/section}
```

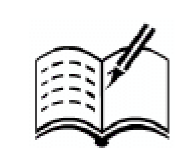
本实例的重点在于讲解如何通过 section 语句循环输出数组中的数据，至于其他内容读者可以参考配套资源中的源程序，这里不做讲解。

技术要点

section 循环语句，用于比较复杂的数组。section 的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```





参数说明：

☑ name：表示循环的名称。
☑ loop：表示循环的数组。
☑ start：表示循环的初始位置，如果 start=2，那么说明循环是从 loop 数组的第 2 个元素开始。
☑ step：表示步长，如果 step=2，那么循环一次后，数组的指针将向下移动两位，以此类推。

Section 循环语句读取的是存储在模板变量中的数组元素，而这个数组元素值是在动态 PHP 文件中通过调用数据库操作类中的$admindb->ExecSQL 方法获取的。

究其根源就是应用 ADODB 类库中的 GetRows()方法获取的查询结果，有关其具体的设置可以参考封装的数据库操作类 AdminDB，该类存储于 system\system.smarty.inc.php 文件中。

实例 246　开启网站注册页面的缓存

（实例位置：配套资源\SL\14\246）

实例说明

缓存的合理应用，应该以实际的开发需要为依据，对于那些需要经常更新的程序是否开启缓存，如果开启缓存，周期长短设置多长时间，这些都必须根据程序的实际需求进行设置。在本实例中开启网站注册页面的缓存，注册页面的运行结果如图 14.7 所示。

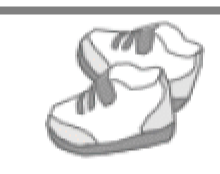


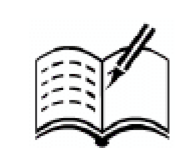
图 14.7　开启网站注册页面的缓存

实现过程

首先创建本实例应用的首页模板，然后创建一个指向模板的处理页，完成用户注册的功能，并且开启用户注册页面的缓存。有关缓存的设置在动态处理页 index.php 中完成，其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );          //设置文件编码格式
require_once("system/system.inc.php");                        //包含配置文件
$smarty->caching=true;                                        //开启缓存
$smarty->cache_lifetime=3600;                                 //设置缓存过期时间
//$smarty->clear_all_cache();                                 //清除所有缓存
if(!$smarty->is_cached('index.html')){
$res=$conn->execute("select * from tb_bccd ");                //执行 select 查询语句
$array=$res->GetArray();
$array_id=array();
$array_name=array();
for($i=0;$i<$res->RecordCount();$i++){
```





参数说明：

- ☑ name：表示循环的名称。
- ☑ loop：表示循环的数组。
- ☑ start：表示循环的初始位置，如果 start=2，那么说明循环是从 loop 数组的第 2 个元素开始。
- ☑ step：表示步长，如果 step=2，那么循环一次后，数组的指针将向下移动两位，以此类推。

Section 循环语句读取的是存储在模板变量中的数组元素，而这个数组元素值是在动态 PHP 文件中通过调用数据库操作类中的$admindb->ExecSQL 方法获取的。

究其根源就是应用 ADODB 类库中的 GetRows()方法获取的查询结果，有关其具体的设置可以参考封装的数据库操作类 AdminDB，该类存储于 system\system.smarty.inc.php 文件中。

实例 246　开启网站注册页面的缓存

（实例位置：配套资源\SL\14\246）

实例说明

缓存的合理应用，应该以实际的开发需要为依据，对于那些需要经常更新的程序是否开启缓存，如果开启缓存，周期长短设置多长时间，这些都必须根据程序的实际需求进行设置。在本实例中开启网站注册页面的缓存，注册页面的运行结果如图 14.7 所示。

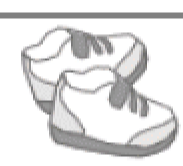


图 14.7　开启网站注册页面的缓存

实现过程

首先创建本实例应用的首页模板，然后创建一个指向模板的处理页，完成用户注册的功能，并且开启用户注册页面的缓存。有关缓存的设置在动态处理页 index.php 中完成，其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );          //设置文件编码格式
require_once("system/system.inc.php");                        //包含配置文件
$smarty->caching=true;                                        //开启缓存
$smarty->cache_lifetime=3600;                                 //设置缓存过期时间
//$smarty->clear_all_cache();                                 //清除所有缓存
if(!$smarty->is_cached('index.html')){
$res=$conn->execute("select * from tb_bccd ");                //执行 select 查询语句
$array=$res->GetArray();
$array_id=array();
$array_name=array();
for($i=0;$i<$res->RecordCount();$i++){
```




```
        $array_id[]=$array[$i][id];
        $array_name[]=$array[$i][name];
    }
    $smarty->assign('cust_id', $array_id);                //设置下拉列表的值
    $smarty->assign('cust_name', $array_name);            //设置列表的显示数据
    }
    $array= explode(' ', mt_rand(1000,9999));             //生成随机验证码
    $smarty->assign('title','开启网站注册页面的缓存');     //将指定数据赋给模板变量
    $smarty->assign('content',$array);                    //将数组赋给模板变量
    $smarty->assign('contents',$arrays);                  //将数组赋给模板变量
    $smarty->display('index.html');                       //指定模板页
    ?>
```

缓存文件存储在 system\Smarty\cache 目录下，而存储路径的设置是在 system\system.smarty.inc.php 文件中完成。有关用户注册的其他功能，请读者参考配套资源中的源程序，这里不再赘述。

技术要点

缓存的设置，既要考虑网站访问速度的提高，又要考虑缓存生命周期的合理性。不能偏执一方，应该采取中庸之道，达到一个最理想的效果。

例如，网站首页是不需要经常更新的内容，缓存的生存周期就可以设置长一些，而类似于论坛中帖子的数据，则可以不开启缓存，因为帖子的数据会不断更新，就没有必要再使用缓存，如果使用缓存可能会导致浏览不到最新的数据。

（1）开启缓存。

开启缓存的方法非常简单，只要将 Smarty 对象中$config 的值设置为 true 即可，同时还要通过 Smarty 对象中的$cache_dir 属性指定缓存文件的存储位置。其操作代码如下：

```
$smarty->caching=true;                                    //开启缓存
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';      //定义缓存文件存储位置
```

（2）设置缓存生命周期。

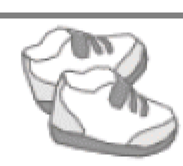
缓存创建成功后，必须为它设置一个生命周期，如果它一直不更新，那么就没有任何意义。设置缓存生命周期应用的是 Smarty 对象中的$cache_lifetime 属性，缓存时间以秒为单位，默认值是 3600 秒。其操作代码如下：

```
$smarty->caching=true;                                    //开启缓存
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';      //定义缓存文件存储位置
$smarty->cache_lifetime=3600                              //设置缓存时间为 1 小时
```

（3）判断模板文件是否已经被缓存。

如果页面已经被缓存，那么就可以直接调用缓存文件，而不再执行动态获取数据和输出的操作。为了避免在开启缓存后，再次执行动态获取数据和输出操作给服务器带来的压力，最佳的方法就是应用 Smarty 对象中的 is_cached()方法，判断指定的模板是否存在缓存，如果存在则直接执行缓存中的文件，否则执行动态获取数据和输出的操作。操作代码如下：

```
$smarty->caching=true;                                    //开启缓存
if(!$smarty->is_cached('index.html')){
```




```
        $array_id[]=$array[$i][id];
        $array_name[]=$array[$i][name];
    }
    $smarty->assign('cust_id', $array_id);                //设置下拉列表的值
    $smarty->assign('cust_name', $array_name);            //设置列表的显示数据
    }
    $array= explode(' ', mt_rand(1000,9999));             //生成随机验证码
    $smarty->assign('title','开启网站注册页面的缓存');      //将指定数据赋给模板变量
    $smarty->assign('content',$array);                    //将数组赋给模板变量
    $smarty->assign('contents',$arrays);                  //将数组赋给模板变量
    $smarty->display('index.html');                       //指定模板页
    ?>
```

缓存文件存储在 system\Smarty\cache 目录下，而存储路径的设置是在 system\system.smarty.inc.php 文件中完成。有关用户注册的其他功能，请读者参考配套资源中的源程序，这里不再赘述。

技术要点

缓存的设置，既要考虑网站访问速度的提高，又要考虑缓存生命周期的合理性。不能偏执一方，应该采取中庸之道，达到一个最理想的效果。

例如，网站首页是不需要经常更新的内容，缓存的生存周期就可以设置长一些，而类似于论坛中帖子的数据，则可以不开启缓存，因为帖子的数据会不断更新，就没有必要再使用缓存，如果使用缓存可能会导致浏览不到最新的数据。

（1）开启缓存。

开启缓存的方法非常简单，只要将 Smarty 对象中$config 的值设置为 true 即可，同时还要通过 Smarty 对象中的$cache_dir 属性指定缓存文件的存储位置。其操作代码如下：

```
$smarty->caching=true;                                        //开启缓存
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';          //定义缓存文件存储位置
```

（2）设置缓存生命周期。

缓存创建成功后，必须为它设置一个生命周期，如果它一直不更新，那么就没有任何意义。设置缓存生命周期应用的是 Smarty 对象中的$cache_lifetime 属性，缓存时间以秒为单位，默认值是 3600 秒。其操作代码如下：

```
$smarty->caching=true;                                        //开启缓存
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';          //定义缓存文件存储位置
$smarty->cache_lifetime=3600                                  //设置缓存时间为 1 小时
```

（3）判断模板文件是否已经被缓存。

如果页面已经被缓存，那么就可以直接调用缓存文件，而不再执行动态获取数据和输出的操作。为了避免在开启缓存后，再次执行动态获取数据和输出操作给服务器带来的压力，最佳的方法就是应用 Smarty 对象中的 is_cached()方法，判断指定的模板是否存在缓存，如果存在则直接执行缓存中的文件，否则执行动态获取数据和输出的操作。操作代码如下：

```
$smarty->caching=true;                                  //开启缓存
if(!$smarty->is_cached('index.html')){
```



```
            //执行动态获取数据和输出的操作
    }
    $smarty->display('index.html');
```

（4）清除模板中缓存。

缓存的清除有两种方法：

第一种是 clear_all_cache()方法，清除所有模板缓存。其语法如下：

```
void clear_all_cache (int expire time)
```

可选参数 expire time，可以指定一个以秒为单位的最小时间，超过这个时间的缓存都将被清除。

第二种是 clear_cache()方法，清除指定模板的缓存。其语法如下：

```
void clear_cache (string template [, string cache id [, string compile id [, int expire time]]])
```

如果这个模板有多个缓存，可以用第二个参数指定要清除缓存的缓存号，还可以通过第三个参数指定编译号。可以把模板分组，以便可以方便地清除一组缓存。第四个参数是可选的，用来指定超过某一时间（以秒为单位）的缓存才会被清除。



实例 247　通过配置文件定义变量

（实例位置：配套资源\SL\14\247）

实例说明

配置文件的应用，有利于设计者管理文件中的模板全局变量。例如，定义一个模板色彩变量。一般情况下如果想改变一个程序的外观色彩，必须更改每一个文件的颜色变量。如果有配置文件，色彩变量就可以保存在一个单独的文件中，只要改变配置文件就可以实现色彩的更新，这与在 Web 页面开发中应用的 CSS 样式非常相似。在本实例中将讲解如何应用 Smarty 中的配置文件，并通过配置文件定义页面中 body 标签的样式，其运行结果如图 14.8 所示。



图 14.8　通过配置文件定义变量

实现过程

具体步骤如下：

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录和配置文件






```
        //执行动态获取数据和输出的操作
    }
    $smarty->display('index.html');
```

（4）清除模板中缓存。

缓存的清除有两种方法：

第一种是 clear_all_cache()方法，清除所有模板缓存。其语法如下：

```
    void clear_all_cache (int expire time)
```

可选参数 expire time，可以指定一个以秒为单位的最小时间，超过这个时间的缓存都将被清除。

第二种是 clear_cache()方法，清除指定模板的缓存。其语法如下：

```
    void clear_cache (string template [, string cache id [, string compile id [, int expire time]]])
```

如果这个模板有多个缓存，可以用第二个参数指定要清除缓存的缓存号，还可以通过第三个参数指定编译号。可以把模板分组，以便可以方便地清除一组缓存。第四个参数是可选的，用来指定超过某一时间（以秒为单位）的缓存才会被清除。

实例 247　通过配置文件定义变量

（实例位置：配套资源\SL\14\247）

实例说明

配置文件的应用，有利于设计者管理文件中的模板全局变量。例如，定义一个模板色彩变量。一般情况下如果想改变一个程序的外观色彩，必须更改每一个文件的颜色变量。如果有配置文件，色彩变量就可以保存在一个单独的文件中，只要改变配置文件就可以实现色彩的更新，这与在 Web 页面开发中应用的 CSS 样式非常相似。在本实例中将讲解如何应用 Smarty 中的配置文件，并通过配置文件定义页面中 body 标签的样式，其运行结果如图 14.8 所示。

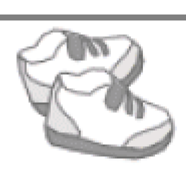


图 14.8　通过配置文件定义变量

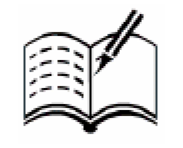
实现过程

具体步骤如下：

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录和配置文件



目录；创建类文件 system.smarty.inc.php，定义 Smarty 的配置类，在该类中指定配置文件存储的目录，定义类的实例化文件 system.inc.php，完成 Smarty 类的实例化操作，并返回操作对象。



（2）创建 index.php 文件，载入类的实例化文件；通过文件系统函数 file_get_contents()读取文本文件中的数据，并且将数据存储到模板变量中，最终指定模板页。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );         //设置文件编码格式
include("system/system.inc.php");                             //包含配置文件
$str = file_get_contents('files/content.txt');                //读取文本文件中的数据
$smarty->assign('content',iconv("gb2312","utf-8",$str));//将文本文件中的数据存储到模板变量中
$smarty->display('index.html');                               //指定模板页
?>
```

（3）创建 index.html 模板页。首先，通过 config_load()函数加载配置文件。然后，应用“#”和$smarty.config 引用配置文件中的变量。最后，通过模板变量输出文本文件中的数据。其关键代码如下：

```
{ config_load file="file_con.conf"}                {* 加载配置文件 *}
<title>{#title#}</title>
<body  leftmargin="{#leftmargin#}"  topmargin="{#topmargin#}"  marginwidth="{#marginwidth#}"
marginheight="{#marginheight#}">
<td height="228" colspan="2" align="left" valign="top" class="{$smarty.config.styles}">{$content}</td>
```

技术要点

（1）创建配置文件。

配置文件可以任意命名，其存储位置由 Smarty 对象的$config_dir 属性指定。如果存在不只在一个区域内使用的变量值，可以使用三引号（"""）将它完整地封装起来。在创建配置文件时，建议在程序运行前使用“#”加一些注释信息，这样有助于程序的阅读、更新。

在配置文件中既可以声明全局变量，也可以声明局部变量。如果声明局部变量，可以使用中括号“[]”括起来，在中括号之内声明的变量属于局部变量，而中括号之外声明的变量都是全局变量。中括号的使用不仅使配置文件中声明变量的模块变得清晰，而且可以在模板中选择加载中括号内的变量。

（2）加载配置文件。

加载配置文件应用 Smarty 的内建函数 config_load()，其语法如下：

```
{config_load file="file_name " section="add_attribute" scope="" global=""}
```

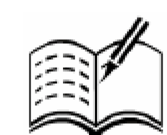
参数说明：

☑ file：指定包含的配置文件的名称。

☑ section：附加属性，当配置文件中包含多个部分时应用，指定具体从哪一部分中取得变量。

☑ scope：加载数据的作用域，取值必须为 local、parent 或 global。local 说明该变量的作用域为当前模板；parent 说明该变量的作用域为当前模板和当前模板的父模板（调用当前模板的模板）；global 说明该变量的作用域为所有模板。





☑　global：说明加载的变量是否全局可见，等同于 scope=parent。

脚下留神：

当指定 scope 属性时，可以设置 global 属性，但模板忽略该属性值，而以 scope 属性为准。

（3）引用配置文件中的变量。

配置文件加载成功后，就可以在模板中引用配置文件中声明的变量。引用配置文件应用的是“#”或者 Smarty 的保留变量$smarty.config。其应用示例如下：

```
{ config_load file="file_con.conf"}                {* 加载配置文件 *}
{#title#}
 <td height="228" colspan="2" align="left" valign="top" class="{$smarty.config.styles}">
```

实例 248　Smarty+ADODB 完成数据的分页显示

（实例位置：配套资源\SL\14\248　视频位置：配套资源\SP\14\248）

实例说明

本实例将介绍通过面向对象的方法完成 Smarty 的配置，ADODB 连接 MySQL、操作 MySQL 数据库以及分页的功能，其运行结果如图 14.9 所示。



图 14.9　Smarty+ADODB 完成数据的分页显示

实现过程

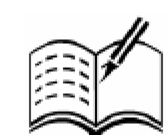
具体步骤如下：

（1）创建 system 文件夹，存储 ADODB 类库和 Smarty 模板文件。创建 system.smarty.inc.php 文件，定义连接数据库、操作数据库和分页类。

数据库连接类 ConnDB，定义构造方法 ConnDB 为成员变量赋值；定义 GetConnId 方法，应用 NewADOConnection()函数连接 MySQL、mssql 或者 Access 数据库，设置数据库编码格式为 UTF-8，最后返回数据库连接对象；定义 CloseConnId 方法，关闭与数据库的连接。

数据库操作类 AdminDB，定义 ExecSQL()方法完成对数据库添加、更新和删除的操作。





☑ global：说明加载的变量是否全局可见，等同于 scope=parent。

脚下留神：

当指定 scope 属性时，可以设置 global 属性，但模板忽略该属性值，而以 scope 属性为准。

（3）引用配置文件中的变量。

配置文件加载成功后，就可以在模板中引用配置文件中声明的变量。引用配置文件应用的是“#”或者 Smarty 的保留变量$smarty.config。其应用示例如下：

```
{ config_load file="file_con.conf"}              {* 加载配置文件 *}
{#title#}
 <td height="228" colspan="2" align="left" valign="top" class="{$smarty.config.styles}">
```

实例 248　Smarty+ADODB 完成数据的分页显示

（实例位置：配套资源\SL\14\248　视频位置：配套资源\SP\14\248）

实例说明

本实例将介绍通过面向对象的方法完成 Smarty 的配置，ADODB 连接 MySQL、操作 MySQL 数据库以及分页的功能，其运行结果如图 14.9 所示。



图 14.9　Smarty+ADODB 完成数据的分页显示

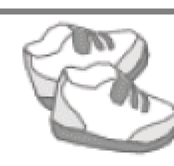
实现过程

具体步骤如下：

（1）创建 system 文件夹，存储 ADODB 类库和 Smarty 模板文件。创建 system.smarty.inc.php 文件，定义连接数据库、操作数据库和分页类。

数据库连接类 ConnDB，定义构造方法 ConnDB 为成员变量赋值；定义 GetConnId 方法，应用 NewADOConnection()函数连接 MySQL、mssql 或者 Access 数据库，设置数据库编码格式为 UTF-8，最后返回数据库连接对象；定义 CloseConnId 方法，关闭与数据库的连接。

数据库操作类 AdminDB，定义 ExecSQL()方法完成对数据库添加、更新和删除的操作。

首先，通过 substr()函数截取 SQL 语句中前 6 个字符串，并将截取的字符串转换成小写。然后，通过 Execute()函数执行 SQL 语句。最后，根据截取的字符串判断 SQL 语句的类型，如果是 select 查询语句，则执行 GetRows()函数，如果查询结果为 0，则返回 false，否则返回查询的数组；如果 SQL 语句为 update、insert 或者 delete 类型，执行成功则返回 true，否则返回 false。

分页类 SepPage，定义 ShowDate()方法完成从数据库中读取数据的操作，执行 PageExecute()函数实现分页功能，并且将返回值定义到数组中；定义 ShowPage()方法根据查询结果，应用分页函数创建分页超链接，并且将返回结果定义到变量$str 中。

Smarty 配置类 SmartyProject，继承 Smarty 类库中 Smarty 类，设置本实例中 Smarty 缓存文件、配置文件、模板文件和编译文件的存储位置。

system.smarty.inc.php 文件的代码请参考本书配套资源，由于篇幅所限这里不再介绍。

（2）创建 system.inc.php 文件，完成对数据库连接、操作、分页以及 Smarty 配置类的实例化操作，其代码如下：

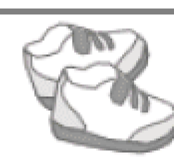
```
<?php
require("system.smarty.inc.php");          //包含配置类文件
$smarty=new SmartyProject();               //实例化配置类
$connobj=new ConnDB("mysql","localhost","root","111","db_database14",false); //数据库连接类实例化
$conn=$connobj->GetConnId();
$admindb=new AdminDB();                    //数据库操作类实例化
$seppage=new SepPage();                    //分页类实例化
?>
```

到此，有关 ADODB 和 Smarty 的安装和配置介绍完毕。

（3）创建 index.php 文件，调用分页类中的 ShowDate()方法，分页读取数据库中的数据，并且将返回值赋给模板变量；调用分页类中的 ShowPage()方法，完成分页超链接的输出，同样将返回值赋给模板变量，最后指定模板页。其代码如下：

```
<?php
require_once("system/system.inc.php");  //包含配置文件
$arraybbs=$seppage->ShowDate("select * from tb_commo where isnew = 1 order by id ",$conn,3,$_GET["page"]);                       //调用分页类，实现分页功能
if(!$arraybbs){
    $smarty->assign("isbbs","F");
}else{
    $smarty->assign("isbbs","T");
  $smarty->assign("showpage",$seppage->ShowPage("商品","个","","a1")); //定义输出分页数据的模板变量 showpage
    $smarty->assign("arr",$arraybbs);
}
  $smarty->assign('title','Smarty+Adodb 完成数据分页显示');
  $smarty->display('index.html');
  ?>
```

（4）创建 index.html 模板文件，应用 section 语句和模板变量完成数据的分页输出。

首先，通过 substr()函数截取 SQL 语句中前 6 个字符串，并将截取的字符串转换成小写。然后，通过 Execute()函数执行 SQL 语句。最后，根据截取的字符串判断 SQL 语句的类型，如果是 select 查询语句，则执行 GetRows()函数，如果查询结果为 0，则返回 false，否则返回查询的数组；如果 SQL 语句为 update、insert 或者 delete 类型，执行成功则返回 true，否则返回 false。

分页类 SepPage，定义 ShowDate()方法完成从数据库中读取数据的操作，执行 PageExecute()函数实现分页功能，并且将返回值定义到数组中；定义 ShowPage()方法根据查询结果，应用分页函数创建分页超链接，并且将返回结果定义到变量$str 中。

Smarty 配置类 SmartyProject，继承 Smarty 类库中 Smarty 类，设置本实例中 Smarty 缓存文件、配置文件、模板文件和编译文件的存储位置。

system.smarty.inc.php 文件的代码请参考本书配套资源，由于篇幅所限这里不再介绍。

（2）创建 system.inc.php 文件，完成对数据库连接、操作、分页以及 Smarty 配置类的实例化操作，其代码如下：

```
<?php
require("system.smarty.inc.php");          //包含配置类文件
$smarty=new SmartyProject();               //实例化配置类
$connobj=new ConnDB("mysql","localhost","root","111","db_database14",false); //数据库连接类实
例化
$conn=$connobj->GetConnId();
$admindb=new AdminDB();                    //数据库操作类实例化
$seppage=new SepPage();                    //分页类实例化
?>
```

到此，有关 ADODB 和 Smarty 的安装和配置介绍完毕。

（3）创建 index.php 文件，调用分页类中的 ShowDate()方法，分页读取数据库中的数据，并且将返回值赋给模板变量；调用分页类中的 ShowPage()方法，完成分页超链接的输出，同样将返回值赋给模板变量，最后指定模板页。其代码如下：

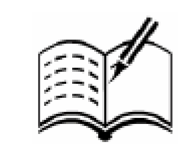
```
<?php
require_once("system/system.inc.php");   //包含配置文件
$arraybbs=$seppage->ShowDate("select * from tb_commo where isnew = 1 order by id ",$conn,3,$_
GET["page"]);                             //调用分页类，实现分页功能
if(!$arraybbs){
    $smarty->assign("isbbs","F");
}else{
    $smarty->assign("isbbs","T");
  $smarty->assign("showpage",$seppage->ShowPage("商品","个","","a1")); //定义输出分页数据的
模板变量 showpage
    $smarty->assign("arr",$arraybbs);
}
  $smarty->assign('title','Smarty+Adodb 完成数据分页显示');
  $smarty->display('index.html');
  ?>
```

（4）创建 index.html 模板文件，应用 section 语句和模板变量完成数据的分页输出。

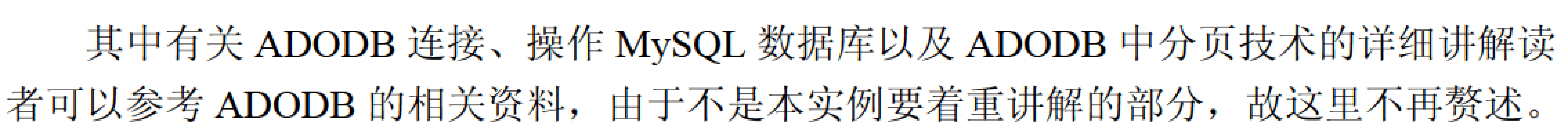


（5）创建 showcommo.html 和 showcommo.php 文件，用于输出指定产品的详细信息。

技术要点



Smarty+ADODB 完成数据的分页显示，其中涵盖了 ADODB 连接 MySQL、ADODB 操作 MySQL 数据库以及 ADODB 中的分页技术，而 Smarty 依旧是将 ADODB 操作 MySQL 获取到的数据通过 assign 方法传递给模板变量，然后在模板页中应用 section 语句循环输出数据。



其中有关 ADODB 连接、操作 MySQL 数据库以及 ADODB 中分页技术的详细讲解读者可以参考 ADODB 的相关资料，由于不是本实例要着重讲解的部分，故这里不再赘述。

在本实例中将 Smarty 的配置方法以及 ADODB 连接和操作数据库的方法都存储在 system.smarty.inc.php 文件中；将 Smarty 配置类、数据库的连接和操作类的实例化存储在 system.inc.php 文件中；将 Smarty 的编译文件、配置文件和缓存文件的存储目录放置在 Smarty 文件夹下；而 Smarty 类库和 ADODB5 类库则存储在与实例根目录的同级目录下，本实例的文件夹架构如图 14.10 所示。

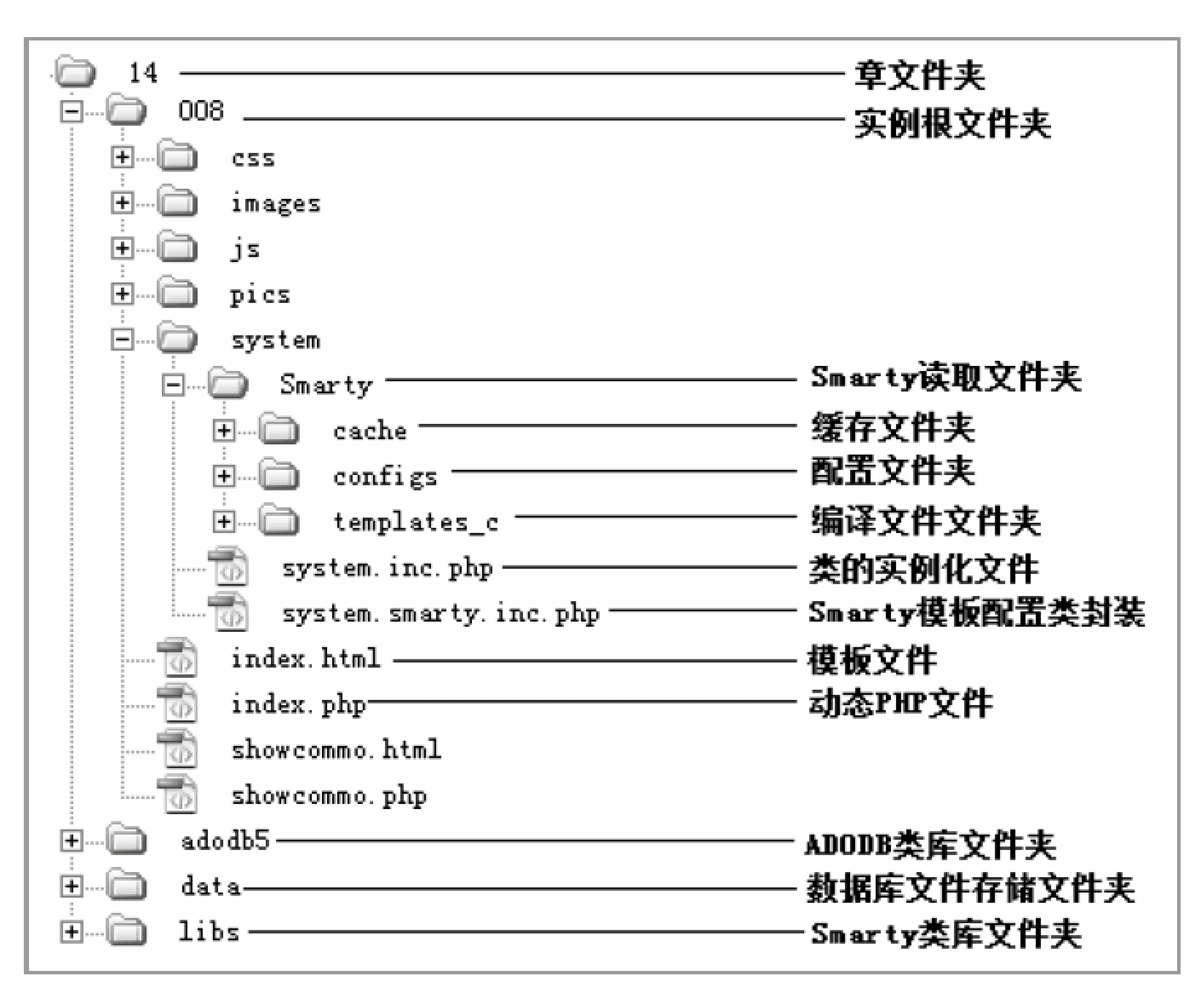


图 14.10　本实例的文件夹架构

实例 249　Smarty 模板中 truncate()方法截取字符串

（实例位置：配套资源\SL\14\249）

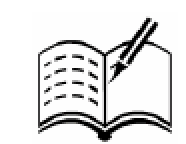
实例说明

在 Smarty 中，提供 truncate()方法对字符串进行截取，该方法会截取到一个词的末尾。在本实例中，应用 truncate()方法对论坛中标题和内容进行截取，并且应用省略号替换截取的内容，运行结果如图 14.11 所示。

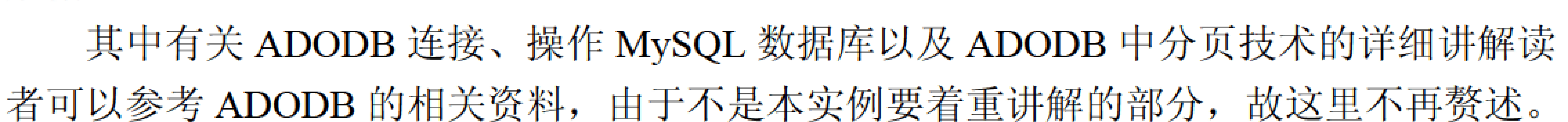


（5）创建 showcommo.html 和 showcommo.php 文件，用于输出指定产品的详细信息。

技术要点



Smarty+ADODB 完成数据的分页显示，其中涵盖了 ADODB 连接 MySQL、ADODB 操作 MySQL 数据库以及 ADODB 中的分页技术，而 Smarty 依旧是将 ADODB 操作 MySQL 获取到的数据通过 assign 方法传递给模板变量，然后在模板页中应用 section 语句循环输出数据。



其中有关 ADODB 连接、操作 MySQL 数据库以及 ADODB 中分页技术的详细讲解读者可以参考 ADODB 的相关资料，由于不是本实例要着重讲解的部分，故这里不再赘述。

在本实例中将 Smarty 的配置方法以及 ADODB 连接和操作数据库的方法都存储在 system.smarty.inc.php 文件中；将 Smarty 配置类、数据库的连接和操作类的实例化存储在 system.inc.php 文件中；将 Smarty 的编译文件、配置文件和缓存文件的存储目录放置在 Smarty 文件夹下；而 Smarty 类库和 ADODB5 类库则存储在与实例根目录的同级目录下，本实例的文件夹架构如图 14.10 所示。

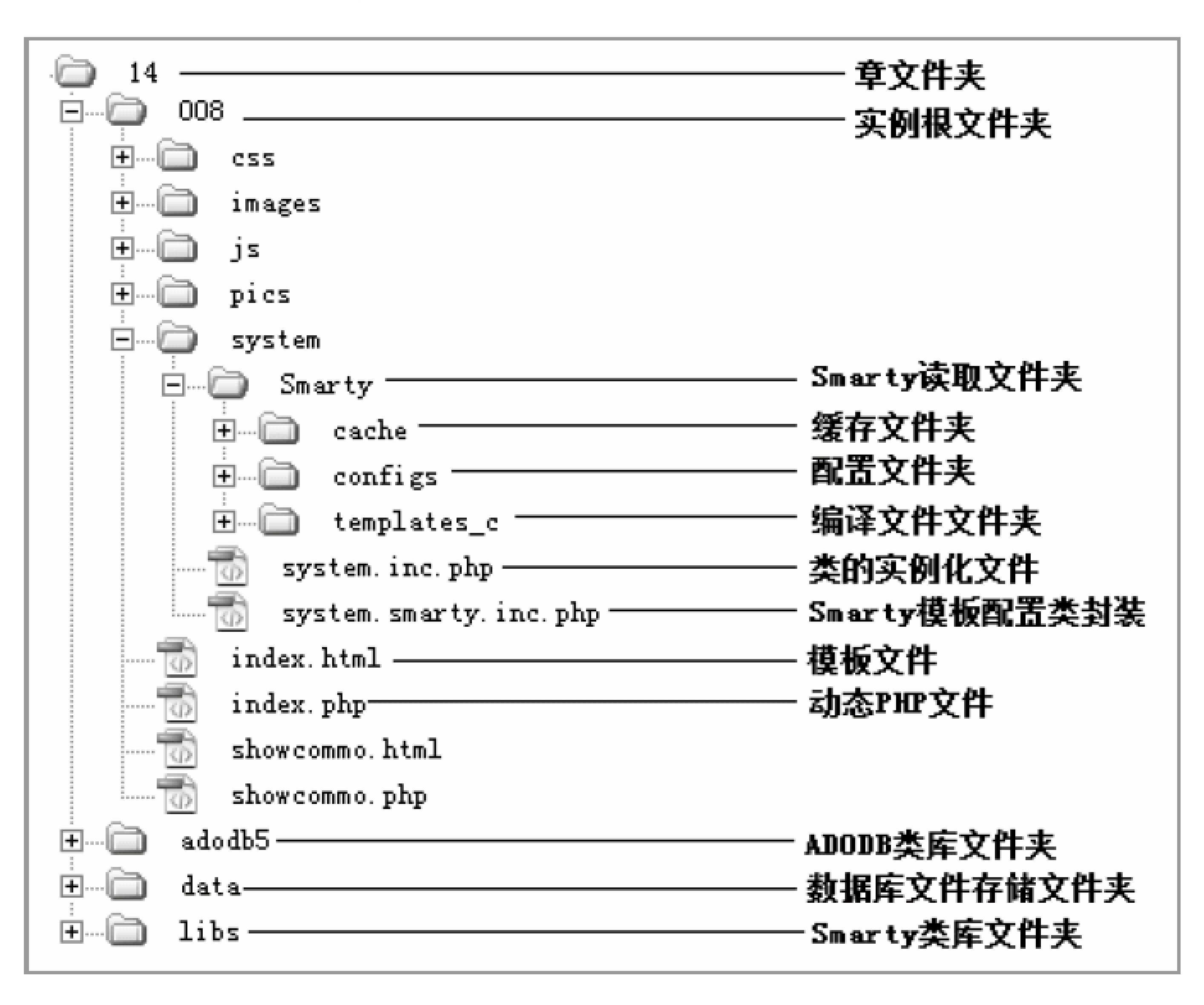


图 14.10　本实例的文件夹架构

实例 249　Smarty 模板中 truncate()方法截取字符串

（实例位置：配套资源\SL\14\249）

实例说明

在 Smarty 中，提供 truncate()方法对字符串进行截取，该方法会截取到一个词的末尾。在本实例中，应用 truncate()方法对论坛中标题和内容进行截取，并且应用省略号替换截取的内容，运行结果如图 14.11 所示。

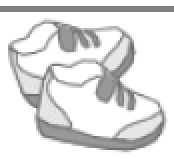





图 14.11　Smarty 模板中 truncate 方法截取字符串

实现过程

具体步骤如下：

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录；创建类文件 system.smarty.inc.php，定义数据库连接、操作、分页和 Smarty 的配置类，定义类的实例化文件 system.inc.php，完成各个类的实例化操作，并返回操作对象。

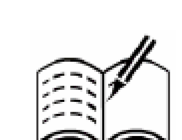
（2）创建 index.php 文件，载入类的实例化文件；分页读取数据库中存储的论坛数据，并且将读取的结果存储到模板变量中，最终指定模板页。其代码如下：

```
<?php
require_once("system/system.inc.php");  //调用指定的文件
$array=$seppage->ShowDate("select * from tb_guestbook order by id desc",$conn,6,$_GET["page"]);
  //调用分页类，实现分页
if(!$array){
  $smarty->assign("iscommo","F");       //判断如果执行失败则输出模板变量 iscommo 的值为 F
}else{
  $smarty->assign("iscommo","T");       //判断如果执行成功，则输出模板变量 iscommo 的值为 T，
  $smarty->assign("showpage",$seppage->ShowPage("帖子","条","","a1")); //定义输出分页数据的
模板变量 showpage
      $smarty->assign("array",$array);
}
  $smarty->assign('title','truncate 方法截取字符串');          //定义标题
  $smarty->display('index.html');                            //指定模板页
?>
```

（3）创建 index.html 模板页，获取模板变量传递的数据，实现数据的分页输出，并且通过 truncate 方法对输出的标题和内容进行截取操作，标题截取 30 个字节，内容截取 60 个字节，应用省略号补齐截取部分。其关键代码如下：

```
{section name=id loop=$array}
<tr>
<td>
{$array[id].id} {$array[id].title|truncate:30:"...":false};
积分：{$array[id].integral}
时间：{$array[id].createtime|truncate:12:""}<br>
            {$array[id].content|truncate:60:"...":false}
</td>
</tr>
{/section}
```





技术要点

truncate 方法，从字符串开始处截取指定长度的字符，默认是 80 个字符。应用示例如下：

```
{$articleTitle|truncate}              <!--截取默认长度的字符串-->
{$articleTitle|truncate:30}               <!--截取 30 个字符-->
{$articleTitle|truncate:30:""}        <!--截取 30 个字符，结尾处默认使用省略号进行添加-->
{$articleTitle|truncate:30:"---"}         <!--截取 30 个字符，结尾处以“---”进行添加-->
{$articleTitle|truncate:30:"":true}       <!--截取 30 个字符串，进行精确截取-->
{$articleTitle|truncate:30:"...":true}    <!--精确截取 30 个字符，以省略号进行添加-->
```

其中$articleTitle 为模板变量。truncate()方法的参数说明如表 14.3 所示。

表 14.3　truncate()方法的参数说明

参　　数	说　　明
1	第一个参数，指定截取字符的数量，类型为 Integer
2	第二个参数，指定截取后追加在截取词后面的字符串，类型为 String
3	第三个参数，类型为 Boolean，如果该值为 false，表示截取到词的边界；如果该值为 true，表示截取时精确到指定字符

实例 250　Register_function()方法注册模板函数

（实例位置：配套资源\SL\14\250）

实例说明

本实例讲解在 Smarty 中注册模板函数的方法，模板函数可以实现与 PHP 中自定义函数相同的功能，其特点是模板函数的注册在动态页中完成，实际的应用在静态页中进行，完全体现 Smarty 的动静分离的开发模式。在本实例分页输出论坛帖子的信息，并通过注册模板函数对帖子的内容进行截取，如果帖子内容超出指定的长度就对其进行截取，并且使用省略号替代被截取部分，运行结果如图 14.12 所示。

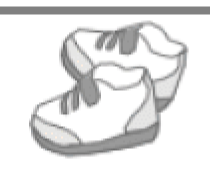


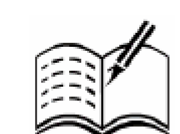
图 14.12　Register function()方法注册模板函数

实现过程

具体步骤如下：

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录；创建类文





Note

技术要点

truncate 方法，从字符串开始处截取指定长度的字符，默认是 80 个字符。应用示例如下：

```
{$articleTitle|truncate}              <!--截取默认长度的字符串-->
{$articleTitle|truncate:30}               <!--截取 30 个字符-->
{$articleTitle|truncate:30:""}        <!--截取 30 个字符，结尾处默认使用省略号进行添加-->
{$articleTitle|truncate:30:"---"}         <!--截取 30 个字符，结尾处以“---”进行添加-->
{$articleTitle|truncate:30:"":true}       <!--截取 30 个字符串，进行精确截取-->
{$articleTitle|truncate:30:"...":true}    <!--精确截取 30 个字符，以省略号进行添加-->
```

其中$articleTitle 为模板变量。truncate()方法的参数说明如表 14.3 所示。

表 14.3　truncate()方法的参数说明

参　数	说　明
1	第一个参数，指定截取字符的数量，类型为 Integer
2	第二个参数，指定截取后追加在截取词后面的字符串，类型为 String
3	第三个参数，类型为 Boolean，如果该值为 false，表示截取到词的边界；如果该值为 true，表示截取时精确到指定字符

实例 250　Register_function()方法注册模板函数

（实例位置：配套资源\SL\14\250）

实例说明

本实例讲解在 Smarty 中注册模板函数的方法，模板函数可以实现与 PHP 中自定义函数相同的功能，其特点是模板函数的注册在动态页中完成，实际的应用在静态页中进行，完全体现 Smarty 的动静分离的开发模式。在本实例分页输出论坛帖子的信息，并通过注册模板函数对帖子的内容进行截取，如果帖子内容超出指定的长度就对其进行截取，并且使用省略号替代被截取部分，运行结果如图 14.12 所示。



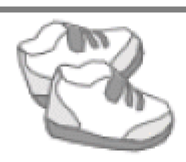
图 14.12　Register function()方法注册模板函数

实现过程

具体步骤如下：

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录；创建类文





件 system.smarty.inc.php，定义数据库连接、操作、分页和 Smarty 的配置类，定义类的实例化文件 system.inc.php，完成各个类的实例化操作，并返回操作对象。

（2）创建 index.php 文件，载入类的实例化文件；编写自定义函数 unHtml，对论坛中的内容进行截取操作；应用 register_function 注册模板函数；分页读取数据库中存储的论坛数据，并且将读取的结果存储到模板变量中，最终指定模板页。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require_once("system/system.inc.php");                              //调用指定的文件
function unHtml($params) {                                          //创建模板函数
  extract($params);                                                 //读取数据
  if(strlen($text)>40){
        $str=mb_substr($text,0,40,"utf-8")."...";                   //截取字符串
  }else{
        $str=$text;
  }
  return $str;                                                      //返回截取结果
}
$smarty->register_function("Util", "unHtml");                       //注册模板函数
$array=$seppage->ShowDate("select * from tb_guestbook order by id desc",$conn,6,$_GET["page"]);
  //调用分页类，实现分页
if(!$array){
        $smarty->assign("iscommo","F"); //判断如果执行失败则输出模板变量 iscommo 的值为 F
}else{
        $smarty->assign("iscommo","T"); //判断如果执行成功，则输出模板变量 iscommo 的值为 T
        $smarty->assign("showpage",$seppage->ShowPage("帖子","条","","a1")); //定义输出分页数
据的模板变量 showpage
      $smarty->assign("array",$array);
}
  $smarty->assign('title','register_function()方法注册模板函数');
  $smarty->display('index.html');

?>
```

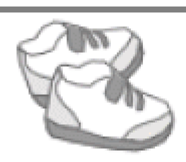
（3）创建 index.html 模板页，获取模板变量传递的数据，实现数据的分页输出，并且调用动态页中注册的模板函数实现对论坛中内容的截取操作。其关键代码如下：

```
{section name=id loop=$array}
<tr>
<td colspan="4" bgcolor="#FFFFFF">{$array[id].id }  {Util text=$array[id].title }积分：
{$array[id].integral}    时间：  {$array[id].createtime}<br>{Util  text=$array[id].
content}</td>
</tr>
{/section}
```

技术要点

（1）register_function()方法，动态注册模板函数插件，其参数说明如表 14.4 所示。语法如下：

```
void register_function (string name, mixed impl, bool cacheable, array or null cache_attrs)
```

件 system.smarty.inc.php，定义数据库连接、操作、分页和 Smarty 的配置类，定义类的实例化文件 system.inc.php，完成各个类的实例化操作，并返回操作对象。

（2）创建 index.php 文件，载入类的实例化文件；编写自定义函数 unHtml，对论坛中的内容进行截取操作；应用 register_function 注册模板函数；分页读取数据库中存储的论坛数据，并且将读取的结果存储到模板变量中，最终指定模板页。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require_once("system/system.inc.php");                              //调用指定的文件
function unHtml($params) {                                          //创建模板函数
  extract($params);                                                 //读取数据
  if(strlen($text)>40){
       $str=mb_substr($text,0,40,"utf-8")."...";                    //截取字符串
  }else{
       $str=$text;
  }
  return $str;                                                      //返回截取结果
}
$smarty->register_function("Util", "unHtml");                       //注册模板函数
$array=$seppage->ShowDate("select * from tb_guestbook order by id desc",$conn,6,$_GET["page"]);
  //调用分页类，实现分页
if(!$array){
       $smarty->assign("iscommo","F"); //判断如果执行失败则输出模板变量 iscommo 的值为 F
}else{
       $smarty->assign("iscommo","T"); //判断如果执行成功，则输出模板变量 iscommo 的值为 T
       $smarty->assign("showpage",$seppage->ShowPage("帖子","条","","a1")); //定义输出分页数
据的模板变量 showpage
     $smarty->assign("array",$array);
}
  $smarty->assign('title','register_function()方法注册模板函数');
  $smarty->display('index.html');

?>
```

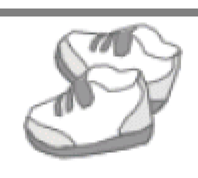
（3）创建 index.html 模板页，获取模板变量传递的数据，实现数据的分页输出，并且调用动态页中注册的模板函数实现对论坛中内容的截取操作。其关键代码如下：

```
{section name=id loop=$array}
<tr>
<td colspan="4" bgcolor="#FFFFFF">{$array[id].id }  {Util text=$array[id].title }积分：
{$array[id].integral}    时间：{$array[id].createtime}<br>{Util text=$array[id].
content}</td>
</tr>
{/section}
```

技术要点

（1）register_function()方法，动态注册模板函数插件，其参数说明如表 14.4 所示。语法如下：

```
void register_function (string name, mixed impl, bool cacheable, array or null cache_attrs)
```



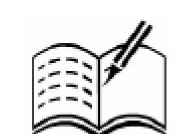


表 14.4 register_function 方法的参数说明

参 数	说 明
name	模板函数的名称
impl	执行函数的名称。执行函数的格式可以是一个包含函数名称的字符串；也可以是一个array(&$object, $method)数组形式，其中&$object 是一个对象的引用，而$method 是它的一个方法；还可以是一个 array(&$ class, $method)数组形式，其中$class 是一个类的名称，$method 是类中的一个方法
cacheable	可选参数，大多数情况下可以省略
cache_attrs	可选参数，大多数情况下可以省略

（2）extract()函数，从数组中将变量导入到当前的符号表。语法如下：

```
int extract ( array var_array [, int extract_type [, string prefix]] )
```

本函数用来将变量从数组中导入到当前的符号表中。接受结合数组 var_array 作为参数并将键名当作变量名，值作为变量的值。对每个键 / 值对都会在当前的符号表中建立变量，并受到 extract_type 和 prefix 参数的影响。

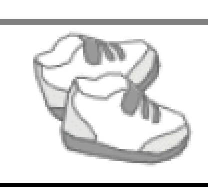
指点迷津：

（1）自 PHP4.0.5 起本函数返回被提取的变量数据。（2）EXTR_IF_EXISTS 和 EXTR_PREFIX_IF_EXISTS 是 PHP4.2.1 中引进的。（3）EXTR_REFS 是 PHP4.3.0 中引进的。

Extract()检查每个键名看是否可以作为一个合法的变量名，同时也检查和符号表中已有的变量名的冲突。对待非法数字和冲突的键名的方法将根据 extract_type 参数决定。可以是以下值之一：

- ☑ EXTR_OVERWRITE：如果有冲突，覆盖已有的变量。
- ☑ EXTR_SKIP：如果有冲突，不覆盖已有的变量。
- ☑ EXTR_PREFIX_SAME：如果有冲突，在变量名前加上前缀 prefix。
- ☑ EXTR_PREFIX_ALL：给所有变量名加上前缀 prefix。自 PHP4.0.5 起这也包括了对数字索引的处理。
- ☑ EXTR_PREFIX_INVALID：仅在非法数字的变量名前加上前缀 prefix。本标记是 PHP 4.0.5 新加的。
- ☑ EXTR_IF_EXISTS：仅在当前符号表中已有同名变量时，覆盖它们的值。其他的都不处理。可以用在已经定义了一组合法的变量，然后要从一个数组中提取值覆盖这些变量的场合，例如$_REQUEST。本标记是 PHP 4.2.0 新加的。
- ☑ EXTR_PREFIX_IF_EXISTS：仅在当前符号表中已有同名变量时，建立附加了前缀的变量名，其他的都不处理。本标记是 PHP 4.2.0 新加的。
- ☑ EXTR_REFS：将变量作为引用提取。这有力地表明了导入的变量仍然引用了 var_array 参数的值。可以单独使用这个标志或者在 extract_type 中用 OR 与其他任何标志结合使用。本标记是 PHP 4.3.0 新加的。

如果没有指定 extract_type，则被假定为 EXTR_OVERWRITE。





指点迷津：

prefix 仅在 extract_type 的值是 EXTR_PREFIX_SAME、EXTR_PREFIX_ALL、EXTR_PREFIX_INVALID 或者 EXTR_PREFIX_IF_EXISTS 时需要，如果附加了前缀后的结果不是合法的变量名，将不会导入到符号表中。

extract()返回成功导入到符号表中的变量数目。

（3）mb_substr()函数，对字符串进行截取操作，并且支持中文字符串的截取。其语法如下：

```
string mb_substr ( string str, int start [, int length [, string encoding]] )
```

mb_substr()函数的参数说明如表 14.5 所示。

表 14.5　mb_substr()函数的参数说明

参　　数	说　　明
str	必选参数，指定操作的字符串
start	必选参数，指定截取的开始位置。参数 start 的指定位置是从 0 开始计算的，即字符串中的第一个字符表示为 0
length	指定截取的字符串长度
encoding	设置字符串的编码格式

实例 251　Smarty 模板中的关键字描红技术

（实例位置：配套资源\SL\14\251　视频位置：配套资源\SP\14\251）

实例说明

在 Smarty 模板中同样可以实现关键字描红的功能，其应用的是 Smarty 中的 replace 变量。在本实例中，将继续应用后台管理系统主页实例的内容，编辑订单查询模块，实现对查询关键字的描红功能，运行结果如图 14.13 所示。

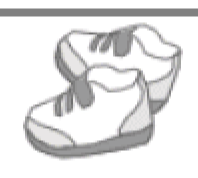


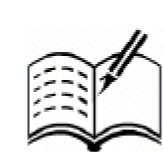
ID	订单号	收货人	时间	操作
1	010603300761	明日科技	2010-08-12 19:50:39	未处理
2	010603300762	明日科技	2010-08-12 19:51:22	已处理
3	010603300763	明日科技	2010-08-12 19:50:39	未处理
4	010603300764	明日科技	2010-08-12 19:51:22	已处理
5	010603300765	明日科技	2010-08-12 19:50:39	未处理
6	010603300766	明日科技	2010-08-12 19:51:22	已处理
7	010603300767	明日科技	2010-08-12 19:50:39	未处理
8	010603300768	明日科技	2010-08-12 19:51:22	已处理

图 14.13　Smarty 模板中的关键字描红

实现过程

本实例首先创建一个 Smarty 模板页，对订单查询模块的内容进行编辑，创建 form 表





单执行查询操作，并且应用 replace 变量将查询结果中的关键字进行描红。

（1）编辑 for_select.html 模板页，创建 form 表单，提交查询的关键字，将数据提交到 for_select.php 页面进行处理；同时在本页中应用 section 语句循环输出查询结果，并且应用 replace 对查询的关键字进行描红操作。其关键代码如下：

```
{section name=id loop=$res}
<tr>
    <td align="center" bgcolor="#FFFFFF">{$res[id].id}</td>
    <td bgcolor="#FFFFFF">{$res[id].number|replace:$select:"<font color='#FF0000'>$select</font>"}
</td>
    <td  bgcolor="#FFFFFF">{$res[id].user|replace:"明日科技":"<font  color='#FF0000'>明日科技
</font>"}</td>
    <td bgcolor="#FFFFFF">{$res[id].dates}</td>
    <td align="center" bgcolor="#FFFFFF">{if $res[id].type==0}未处理{else}已处理{/if}</td>
</tr>
{/section}
```

（2）编辑 for_select.php 动态页，以 form 表单中提交的关键字为条件，执行模糊查询，如果查询结果为真，则将查询结果赋给模板变量，同时将查询的关键字也赋给指定的模板变量；如果查询结果为 false，则将为模板变量赋值 F。其代码如下：

```
<?php
$smarty->assign(title,"Smarty 模板中的关键字描红技术");      //定义标题变量
if($_POST['number']!=""){                                   //判断查询关键字是否为空
  $sql="select * from tb_order where number like '%".$_POST['number']."%'";//定义模糊查询的语句
  $res=$admindb->ExecSQL($sql,$conn);                       //执行模糊查询
  if($res){
        $smarty->assign("select",$_POST['number']);         //将查询的关键字赋给模板变量
        $smarty->assign("boo","T");                         //为模板变量赋值为 T
        $smarty->assign("res",$res);                        //将查询结果赋给模板变量
  }else{
        $smarty->assign("boo","F");
  }
}else{
  $smarty->assign("boo","F");
}
?>
```

技术要点

（1）Smarty 模板中的关键字描红应用的是 replace 变量，其实现字符串的简单查询和替换操作。其包括两个参数：第一个参数是被替换的文本字符串；第二个参数是用来替换的文本字符串。其应用示例如下：

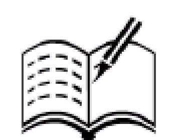
```
{$articleTitle|replace:"Garden":"Vineyard"}
```

在这个示例中，将模板变量$articleTitle 中的 Garden 替换为 Vineyard。

（2）查询关键字描红的原理：在 PHP 动态页中获取 form 表单提交的查询关键字，并且执行模糊查询，将查询结果赋给模板变量，同时将查询的关键字也赋给模板变量，然后



在模板页中，通过 section 语句循环输出查询结果，并应用 replace 将查询结果中关键字的字体设置为红色。



实例 252　Smarty 模板中生成数字验证码

（实例位置：配套资源\SL\14\252　视频位置：配套资源\SP\14\252）

实例说明

本实例应用 Smarty 模板中的 foreach 语句在模板页中直接生成一个数字验证码，它相对于在动态 PHP 中开发的验证码更加简单、实用。Smarty 模板中生成的验证码运行结果如图 14.14 所示。



图 14.14　Smarty 模板中生成数字验证码

实现过程

具体步骤如下：

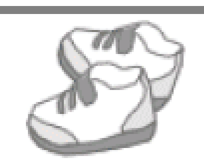
（1）创建 system 文件夹，封装 Smarty 模板的配置方法，这里不再赘述。

（2）创建 index.php 文件，生成验证码，并且将验证码赋给模板变量，最后指定模板页。其代码如下：

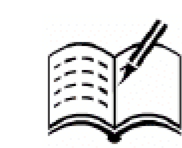
```
<?php
require_once("system/system.inc.php");                    //包含配置文件
$array= explode(' ', mt_rand(1000,9999));                 //生成随机验证码
$smarty->assign('title','Smarty 模板中生成数字验证码');    //将指定数据赋给模板变量
$smarty->assign('content',$array);                        //将数组赋给模板变量
$smarty->display('index.html');                           //指定模板页
?>
```

（3）创建 index.html 模板文件，设计用户登录页面，添加用户登录的表单元素。同时应用 foreach 语句输出模板变量中传递的验证码。其关键代码如下：

```
<form id="form1" name="form1" method="post" action="index_ok.php">
        <input name="user" type="text" id="user" size="10" />
      <input type="hidden" name="checks" value="{foreach key=key item=item from=$content}
{$item} {/foreach}" />
        <input name="check" type="text" size="8" />
```



```
                <input name="pass" type="password" id="pass" size="10" />
                <input type="image" name="imageField" src="images/Blog_03_03.jpg" />
                <input type="image" name="imageField2" onclick="form.reset();return false;" src="images/
Blog_03_06.jpg" />
        </form>
```



（4）创建 index_ok.php 文件，获取表单中提交的数据，对用户登录信息进行验证。如果用户名和密码正确则跳转到 main.php 页面，否则跳转到 index.php 页面。

（5）创建 main.php 和 main.html 文件，作为网站的主页。

技术要点

严格地说这个验证码不是在 Smarty 模板页中生成的，只是通过 foreach 语句将 PHP 动态页中生成的验证码在模板页中输出。因此本实例开发的关键是 foreach 语句的应用和验证码的生成。

（1）应用 mt_rand()函数生成验证码。mt_rand()函数的语法如下：

```
int mt_rand ( [int min, int max])
```

如果 mt_rand()函数没有提供可选参数 min 和 max，则返回 0 到 RAND_MAX 之间的伪随机数。

（2）应用 explode()函数将生成的验证码写入到数组中。explode()函数的语法如下：

```
array explode(string separator, string string, [int limit])
```

返回由字符串组成的数组，每个元素都是 string 的一个子串，它们被字符串 separator 作为边界点分隔出来。如果设置 limit 参数，则返回的数组包含最多 limit 个元素，而最后那个元素将包含 string 的剩余部分；如果 separator 为空字符串（""），explode()函数将返回 false；如果 separator 所包含的值在 string 中找不到，那么 explode()函数将返回包含 string 单个元素的数组；如果参数 limit 是负数，则返回除了最后的-limit 个元素外的所有元素。

（3）通过 assign()方法将验证码数组赋给模板变量。

（4）应用 foreach 语句循环输出模板变量中数组元素。foreach 语法如下：

```
{foreach name=foreach_name key=key item=item from=arr_name}
…
{/foreach}
```

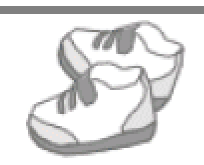
参数说明：

☑ name：该循环的名称。

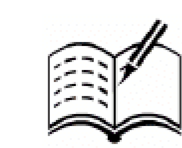
☑ key：当前元素的键值。

☑ item：必选参数，当前元素的变量名。

☑ from：必选参数，该循环的数组。



```
                <input name="pass" type="password" id="pass" size="10" />
                <input type="image" name="imageField" src="images/Blog_03_03.jpg" />
                <input type="image" name="imageField2" onclick="form.reset();return false;" src="images/
Blog_03_06.jpg" />
        </form>
```



（4）创建 index_ok.php 文件，获取表单中提交的数据，对用户登录信息进行验证。如果用户名和密码正确则跳转到 main.php 页面，否则跳转到 index.php 页面。

（5）创建 main.php 和 main.html 文件，作为网站的主页。

技术要点

严格地说这个验证码不是在 Smarty 模板页中生成的，只是通过 foreach 语句将 PHP 动态页中生成的验证码在模板页中输出。因此本实例开发的关键是 foreach 语句的应用和验证码的生成。

（1）应用 mt_rand()函数生成验证码。mt_rand()函数的语法如下：

```
int mt_rand ( [int min, int max])
```

如果 mt_rand()函数没有提供可选参数 min 和 max，则返回 0 到 RAND_MAX 之间的伪随机数。

（2）应用 explode()函数将生成的验证码写入到数组中。explode()函数的语法如下：

```
array explode(string separator, string string, [int limit])
```

返回由字符串组成的数组，每个元素都是 string 的一个子串，它们被字符串 separator 作为边界点分隔出来。如果设置 limit 参数，则返回的数组包含最多 limit 个元素，而最后那个元素将包含 string 的剩余部分；如果 separator 为空字符串（""），explode()函数将返回 false；如果 separator 所包含的值在 string 中找不到，那么 explode()函数将返回包含 string 单个元素的数组；如果参数 limit 是负数，则返回除了最后的-limit 个元素外的所有元素。

（3）通过 assign()方法将验证码数组赋给模板变量。

（4）应用 foreach 语句循环输出模板变量中数组元素。foreach 语法如下：

```
{foreach name=foreach_name key=key item=item from=arr_name}
…
{/foreach}
```

参数说明：

☑ name：该循环的名称。

☑ key：当前元素的键值。

☑ item：必选参数，当前元素的变量名。

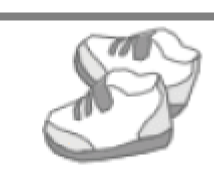
☑ from：必选参数，该循环的数组。



ThinkPHP 框架

本章读者可以学到如下实例：

- ⏭ 实例 253　用户信息的查询、更新和删除
- ⏭ 实例 254　用户登录和数据的分页输出
- ⏭ 实例 255　通过验证码类和分页类完成用户登录和分页输出
- ⏭ 实例 256　通过 ThinkPHP 中的扩展类生成中文验证码
- ⏭ 实例 257　可以传递查询条件的分页
- ⏭ 实例 258　应用 ThinkPHP 中的扩展类上传文件



实例 253　用户信息的查询、更新和删除

（实例位置：配套资源\SL\15\253　视频位置：配套资源\SP\15\253）

实例说明

本实例应用 ThinkPHP 中的 CURD 操作，实现对用户信息的查询、更新和删除操作。其运行效果如图 15.1 所示。

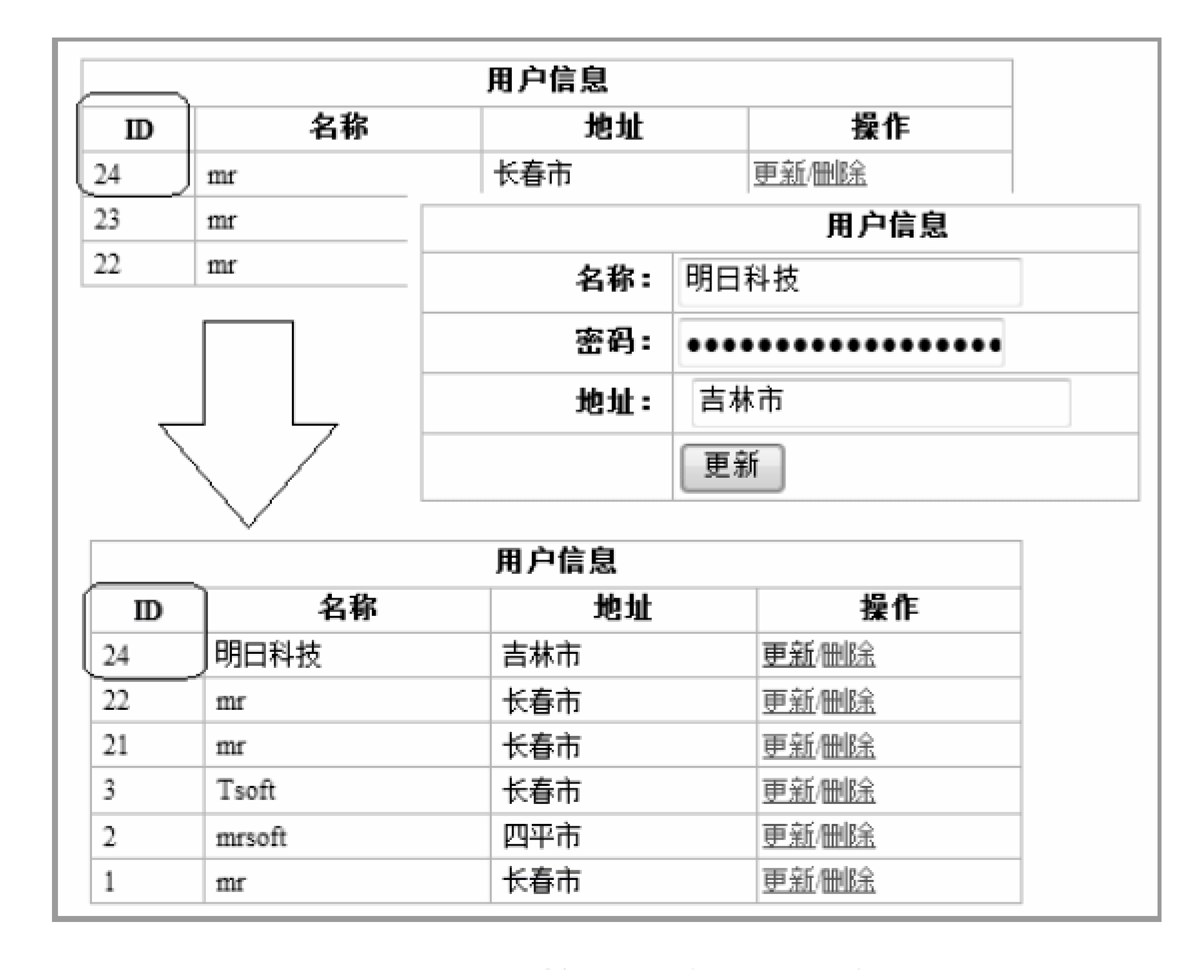


图 15.1　数据更新和删除

实现过程

具体步骤如下：

（1）创建 253 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。

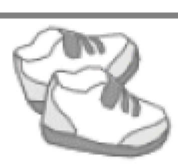
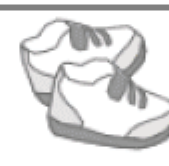
（2）在 253 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');            //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                     //定义项目名称
define('APP_PATH', './App');                   //定义项目路径
require(THINK_PATH."/ThinkPHP.php");           //加载框架入口文件
App::run();                                    //实例化一个网站应用实例
?>
```

（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
```

```
        'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
        'DB_USER'=>'root',
        'DB_PWD'=>'111',
        'DB_PREFIX'=>'think_',
        // 其他项目配置参数……
        'APP_DEBUG' => false,            //关闭调试模式
);
?>
```

（5）定位到 App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，以记录的 ID 值为条件，降幂循环输出 10 条记录。其代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");            //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        $db = M('User');                         //实例化模型类，参数数据表名称，不包含前缀
        $select = $db->order('id desc')->limit(10)->select();
        $this->assign('select',$select);                  //模板变量赋值
        $this->display();                                 //指定模板页
  }
```

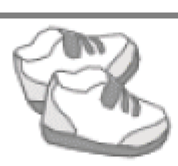
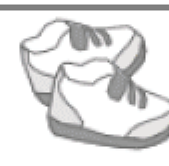
定义 update()方法，首先根据超链接传递 ID 值执行查询，查询出指定的数据，并且将查询结果赋给指定的模板变量。然后，判断表单提交的 ID 值是否存在，如果存在则以 ID 为条件，对指定的数据进行更新操作。其关键代码如下：

```
  public function update(){
        $db = M('User');                         //实例化模型类，参数数据表名称，不包含前缀
        $select = $db->where('id='.$_GET['id'])->select();
        $this->assign('select',$select);                  //模板变量赋值
        $this->display(update);                           //指定模板页
        if(isset($_POST['id'])){
              $data['user'] = $_POST['user'];             //要修改的数据对象属性赋值
              $data['pass'] = md5($_POST['pass']);
              $data['address'] = $_POST['address'];
              $result=$db->where('id='.$_POST['id'])->save($data);     //根据条件保存修改的数据
              if($result){
                    $this->redirect('Index/index','', 2,'数据更新成功');  //页面重定向
              }
        }

  }
```

定义 delete()方法，根据超链接传递的 ID 值，删除数据库中指定的记录。其关键代码如下：

```
  public function delete(){
        $db = M('User');                         //实例化模型类，参数数据表名称，不包含前缀
        $result=$db->where('id='.$_GET['id'])->delete();              //删除 id 为 5 的用户数据
```

```
        'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
        'DB_USER'=>'root',
        'DB_PWD'=>'111',
        'DB_PREFIX'=>'think_',
        // 其他项目配置参数……
        'APP_DEBUG' => false,           //关闭调试模式
    );
    ?>
```

（5）定位到 App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，以记录的 ID 值为条件，降幂循环输出 10 条记录。其代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");           //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        $db = M('User');                        //实例化模型类，参数数据表名称，不包含前缀
        $select = $db->order('id desc')->limit(10)->select();
        $this->assign('select',$select);                  //模板变量赋值
        $this->display();                                 //指定模板页
  }
```

定义 update()方法，首先根据超链接传递 ID 值执行查询，查询出指定的数据，并且将查询结果赋给指定的模板变量。然后，判断表单提交的 ID 值是否存在，如果存在则以 ID 为条件，对指定的数据进行更新操作。其关键代码如下：

```
  public function update(){
        $db = M('User');                        //实例化模型类，参数数据表名称，不包含前缀
        $select = $db->where('id='.$_GET['id'])->select();
        $this->assign('select',$select);                  //模板变量赋值
        $this->display(update);                           //指定模板页
        if(isset($_POST['id'])){
              $data['user'] = $_POST['user'];             //要修改的数据对象属性赋值
              $data['pass'] = md5($_POST['pass']);
              $data['address'] = $_POST['address'];
              $result=$db->where('id='.$_POST['id'])->save($data);    //根据条件保存修改的数据
              if($result){
                    $this->redirect('Index/index','', 2,'数据更新成功');  //页面重定向
              }
        }

  }
```

定义 delete()方法，根据超链接传递的 ID 值，删除数据库中指定的记录。其关键代码如下：

```
  public function delete(){
        $db = M('User');                        //实例化模型类，参数数据表名称，不包含前缀
        $result=$db->where('id='.$_GET['id'])->delete();          //删除 id 为 5 的用户数据
```



```
                if($result){
                    $this->redirect('Index/index','', 2,'数据删除成功');          //页面重定向
                }
            }
        }
        ?>
```



（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，应用 ThinkPHP 内置模板引擎中的 foreach 标签循环输出模板变量传递的数据；创建更新和删除超链接，将指定记录的 ID 作为参数进行传递。其关键代码如下：

```
<foreach name='select' item='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
    <td bgcolor="#FFFFFF"><a href="__URL__/update?id={$user.id}">更新</a>/<a href="__URL
__/delete?id={$user.id}">删除</a></td>
  </tr>
  </foreach>
```

（7）在 Index 模块文件夹下，编辑 update.html 模板文件，创建表单，将从模板变量中读取的数据作为表单元素的默认值进行输出，将表单中数据提交到控制器的 update()方法中完成数据的更新操作。其关键代码如下：

```
<form id="form2" name="form2" method="post" action="__URL__/update">
<table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor=
"#FFFFFF">
<foreach name='select' item='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF" class="right" width="103">名称：</td>
    <td bgcolor="#FFFFFF" width="289"> <input type="hidden" name="id" id="hiddenField"
value="{$user.id}" /><input name="user" type="text" id="user" size="20" value="{$user.user}" /></td>
    </tr>
  <tr class="content">
    <td bgcolor="#FFFFFF" class="right">密码：</td>
    <td bgcolor="#FFFFFF"><input name="pass" type="password" id="pass" size="20"
value="{$user.pass}" />
     </td>
    </tr>
  <tr class="content">
    <td bgcolor="#FFFFFF" class="right"> 地址：</td>
    <td bgcolor="#FFFFFF"> 
      <input name="address" type="text" id="address" size="30" value="{$user.address}" />
    </td>
    </tr>
  <tr class="content">
    <td bgcolor="#FFFFFF"><input type="submit" name="button" id="button" value=" 更 新 "
/></td>
```



```
                if($result){
                    $this->redirect('Index/index','', 2,'数据删除成功');        //页面重定向
                }
            }
        }
        ?>
```

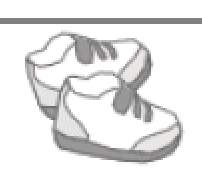


（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，应用 ThinkPHP 内置模板引擎中的 foreach 标签循环输出模板变量传递的数据；创建更新和删除超链接，将指定记录的 ID 作为参数进行传递。其关键代码如下：

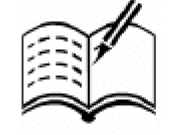
```
        <foreach name='select' item='user' >
          <tr class="content">
            <td bgcolor="#FFFFFF"> {$user.id}</td>
            <td bgcolor="#FFFFFF"> {$user.user}</td>
            <td bgcolor="#FFFFFF"> {$user.address}</td>
            <td bgcolor="#FFFFFF"><a href="__URL__/update?id={$user.id}">更新</a>/<a href="__URL
__/delete?id={$user.id}">删除</a></td>
          </tr>
          </foreach>
```

（7）在 Index 模块文件夹下，编辑 update.html 模板文件，创建表单，将从模板变量中读取的数据作为表单元素的默认值进行输出，将表单中数据提交到控制器的 update()方法中完成数据的更新操作。其关键代码如下：

```
        <form id="form2" name="form2" method="post" action="__URL__/update">
        <table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor=
"#FFFFFF">
        <foreach name='select' item='user' >
          <tr class="content">
            <td bgcolor="#FFFFFF" class="right" width="103">名称：</td>
            <td bgcolor="#FFFFFF" width="289"> <input type="hidden" name="id" id="hiddenField"
value="{$user.id}" /><input name="user" type="text" id="user" size="20" value="{$user.user}" /></td>
            </tr>
          <tr class="content">
            <td bgcolor="#FFFFFF" class="right">密码：</td>
            <td bgcolor="#FFFFFF"><input name="pass" type="password" id="pass" size="20"
value="{$user.pass}" />
              </td>
            </tr>
          <tr class="content">
            <td bgcolor="#FFFFFF" class="right"> 地址：</td>
            <td bgcolor="#FFFFFF"> 
              <input name="address" type="text" id="address" size="30" value="{$user.address}" />
            </td>
            </tr>
          <tr class="content">
            <td bgcolor="#FFFFFF"><input type="submit" name="button" id="button" value="更新"
/></td>
```



```
        </tr>
        </foreach>
    </table>
    </form>
```



Note

技术要点

本章中所举实例都是应用 ThinkPHP 框架进行开发，巩固读者对 ThinkPHP 框架技术的理解，增强应用 ThinkPHP 框架开发实战项目的能力。

本实例中的技术要点可以归纳为如下几个方面：

（1）通过 M 快捷方法实例化基础模型类。在没有定义任何模型的时候，可以使用下面的方法实例化一个模型类来进行操作：

```
$User = new Model('User');
$User->select();          //进行其他的数据操作
```

或者使用 M 快捷方法进行实例化，其效果是相同的。

```
$User = M('User');
$User->select();          //进行其他的数据操作
```

这种方法最简单高效，因为不需要定义任何的模型类，所以支持跨项目调用。缺点也是因为没有自定义的模型类，因此无法写入相关的业务逻辑，只能完成基本的 CURD 操作。

（2）连贯操作与 CURD 操作配合应用，完成数据的创建、更新、读取和删除。

① Where：用于查询或者更新条件的定义。参数支持字符串、数组和对象。

② Order：对结果进行排序。排序方法支持对多个字段的排序，例如，order('status desc,id asc')。order()方法的参数支持字符串和数组，数组的用法如下：order(array ('status'=>'desc','id'))。

③ Limit：结果限制。在 ThinkPHP 中，无论操作的是 MySQL、MS SQL Server 还是 Oracle 数据库，其 limit()的方法是统一的，即 limit('offset,length')。例如，limit ('1,10')。获取从第一条记录开始的 10 条记录。

④ 查询：select()方法的返回值是一个二维数组，如果没有查询到任何结果的话，也是返回一个空的数组。配合上面提到的连贯操作方法可以完成复杂的数据查询。

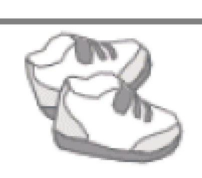
⑤ 更新：save()方法在执行更新数据的操作时，如果没有设置任何更新条件，且数据对象本身也不包含主键字段，那么 save()方法不会更新任何数据库的记录。

⑥ 删除：delete()方法可以用于删除单个或者多个数据，主要取决于删除条件，也就是 where()方法的参数，也可以用 order()和 limit()方法来限制要删除的个数，例如，删除所有状态为 0 的 5 个用户数据按照创建时间排序。代码如下：

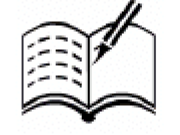
```
$User = M("User");                    // 实例化 User 对象
$User->where('status=0')->order('create_time')->limit('5')->delete();
```

（3）在控制器中通过 assign()方法为模板变量赋值，通过 display()指定模板页，通过 redirect()方法完成页面的重定向。

① 通过 assign()方法将控制器中获取的数据赋给模板变量。例如，$this->assign('name',



```
        </tr>
      </foreach>
    </table>
    </form>
```



Note

技术要点

本章中所举实例都是应用 ThinkPHP 框架进行开发，巩固读者对 ThinkPHP 框架技术的理解，增强应用 ThinkPHP 框架开发实战项目的能力。

本实例中的技术要点可以归纳为如下几个方面：

（1）通过 M 快捷方法实例化基础模型类。在没有定义任何模型的时候，可以使用下面的方法实例化一个模型类来进行操作：

```
$User = new Model('User');
$User->select();            //进行其他的数据操作
```

或者使用 M 快捷方法进行实例化，其效果是相同的。

```
$User = M('User');
$User->select();            //进行其他的数据操作
```

这种方法最简单高效，因为不需要定义任何的模型类，所以支持跨项目调用。缺点也是因为没有自定义的模型类，因此无法写入相关的业务逻辑，只能完成基本的 CURD 操作。

（2）连贯操作与 CURD 操作配合应用，完成数据的创建、更新、读取和删除。

① Where：用于查询或者更新条件的定义。参数支持字符串、数组和对象。

② Order：对结果进行排序。排序方法支持对多个字段的排序，例如，order('status desc,id asc')。order()方法的参数支持字符串和数组，数组的用法如下：order(array ('status'=>'desc','id'))。

③ Limit：结果限制。在 ThinkPHP 中，无论操作的是 MySQL、MS SQL Server 还是 Oracle 数据库，其 limit()的方法是统一的，即 limit('offset,length')。例如，limit ('1,10')。获取从第一条记录开始的 10 条记录。

④ 查询：select()方法的返回值是一个二维数组，如果没有查询到任何结果的话，也是返回一个空的数组。配合上面提到的连贯操作方法可以完成复杂的数据查询。

⑤ 更新：save()方法在执行更新数据的操作时，如果没有设置任何更新条件，且数据对象本身也不包含主键字段，那么 save()方法不会更新任何数据库的记录。

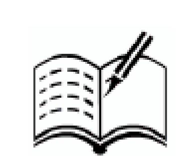
⑥ 删除：delete()方法可以用于删除单个或者多个数据，主要取决于删除条件，也就是 where()方法的参数，也可以用 order()和 limit()方法来限制要删除的个数，例如，删除所有状态为 0 的 5 个用户数据按照创建时间排序。代码如下：

```
$User = M("User");                          //  实例化 User 对象
$User->where('status=0')->order('create_time')->limit('5')->delete();
```

（3）在控制器中通过 assign()方法为模板变量赋值，通过 display()指定模板页，通过 redirect()方法完成页面的重定向。

① 通过 assign()方法将控制器中获取的数据赋给模板变量。例如，$this->assign('name',





$value);。

② 通过 display()方法指定模板页（display('操作名')）。调用当前模块的其他操作模板。例如，当前是 User 模块下面的 read 操作，而要调用 User 模块的 edit 操作模版，$this->display('edit');。

③ 通过 redirect()方法实现页面的重定向。定义规则分为如下三种：第一种基本规则指定项目和模块，第二种数组传参数，第三种不指定项目、模块，即表示当前项目和模块。

```
redirect('[项目://][路由@][分组名-模块/]操作? 参数 1=值 1[&参数 N=值 N]')
redirect('[项目://][路由@][分组名-模块/]操作',array('参数 1'=>'值 1' [,'参数 N'=>'值 N']))
$this->redirect('Index/index','', 5,'页面跳转中');//停留 5 秒，跳到 Index 模块 index 操作，显示页面
跳转字样
```

（4）应用内置模板引擎（ThinkTemplate）中的 foreach 标签在模板页中循环输出模板变量传递的数据。foreach 标签的语法如下：

```
<foreach name="list" item="vo" >
  {$vo.id}
  {$vo.name}
</foreach>
```

foreach 标签用于循环输出，它比 volist 标签简洁，没有 volist 标签那么多的功能。优势是可以对对象进行遍历输出，而 volist 标签通常是用于输出数组。

实例 254　用户登录和数据的分页输出

（实例位置：配套资源\SL\15\254　视频位置：配套资源\SP\15\254）

实例说明

本实例实现用户登录功能，将登录用户的信息存储到 SESSION 变量中，应用 ThinkPHP 中提供的分页扩展类和 Page()方法完成数据的分页输出。其运行效果如图 15.2 所示。



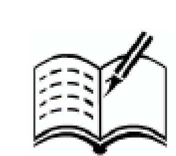
图 15.2　用户登录和数据的分页输出

实现过程

具体步骤如下：

（1）创建 002 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。





$value);。

② 通过 display()方法指定模板页（display('操作名')）。调用当前模块的其他操作模板。例如，当前是 User 模块下面的 read 操作，而要调用 User 模块的 edit 操作模版，$this->display('edit');。

③ 通过 redirect()方法实现页面的重定向。定义规则分为如下三种：第一种基本规则指定项目和模块，第二种数组传参数，第三种不指定项目、模块，即表示当前项目和模块。

```
redirect('[项目://][路由@][分组名-模块/]操作? 参数 1=值 1[&参数 N=值 N]')
redirect('[项目://][路由@][分组名-模块/]操作',array('参数 1'=>'值 1' [,'参数 N'=>'值 N']))
$this->redirect('Index/index','', 5,'页面跳转中');//停留 5 秒，跳到 Index 模块 index 操作，显示页面跳转字样
```

（4）应用内置模板引擎（ThinkTemplate）中的 foreach 标签在模板页中循环输出模板变量传递的数据。foreach 标签的语法如下：

```
<foreach name="list" item="vo" >
  {$vo.id}
  {$vo.name}
</foreach>
```

foreach 标签用于循环输出，它比 volist 标签简洁，没有 volist 标签那么多的功能。优势是可以对对象进行遍历输出，而 volist 标签通常是用于输出数组。

实例 254　用户登录和数据的分页输出

（实例位置：配套资源\SL\15\254　视频位置：配套资源\SP\15\254）

实例说明

本实例实现用户登录功能，将登录用户的信息存储到 SESSION 变量中，应用 ThinkPHP 中提供的分页扩展类和 Page()方法完成数据的分页输出。其运行效果如图 15.2 所示。

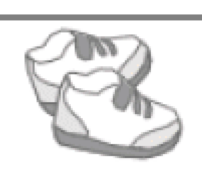


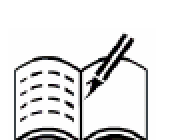
图 15.2　用户登录和数据的分页输出

实现过程

具体步骤如下：

（1）创建 002 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。





Note

（2）在 002 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');            //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                      //定义项目名称
define('APP_PATH', './App');                    //定义项目路径
require(THINK_PATH."/ThinkPHP.php");            //加载框架入口文件
App::run();                                     //实例化一个网站应用实例
?>
```

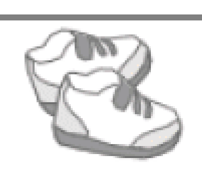
（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

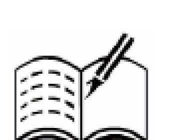
（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                         //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                        //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");        //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
            if(isset($_POST['user']) && isset($_POST['pass'])){
                $db = M();                      //实例化模型类，参数数据表名称，不包含前缀
                $select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                    //执行查询语句，验证用户名和密码是否正确
                if($select){
                    $_SESSION['admin']=$_POST['user'];  //将登录用户名存储到 SESSION 中
                    $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');//页面
重定向
                }else{
                    $this->redirect('Index/index','', 2,'用户名或者密码不正确！'); //页面重定向
                }
            }else{
                $this->redirect('Index/index','', 2,'用户名、密码不能为空！');        //页面重定向
            }
```





（2）在 002 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');          //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                    //定义项目名称
define('APP_PATH', './App');                  //定义项目路径
require(THINK_PATH."/ThinkPHP.php");          //加载框架入口文件
App::run();                                   //实例化一个网站应用实例
?>
```

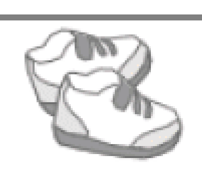
（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

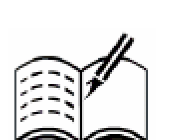
（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                       //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                        //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");        //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
              if(isset($_POST['user']) && isset($_POST['pass'])){
                    $db = M();                  //实例化模型类，参数数据表名称，不包含前缀
                    $select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                       //执行查询语句，验证用户名和密码是否正确
                    if($select){
                          $_SESSION['admin']=$_POST['user'];  //将登录用户名存储到 SESSION 中
                          $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');//页面
重定向
                    }else{
                          $this->redirect('Index/index','', 2,'用户名或者密码不正确！'); //页面重定向
                    }
              }else{
                    $this->redirect('Index/index','', 2,'用户名、密码不能为空！');       //页面重定向
              }
```





（2）在 002 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');          //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                    //定义项目名称
define('APP_PATH', './App');                  //定义项目路径
require(THINK_PATH."/ThinkPHP.php");          //加载框架入口文件
App::run();                                   //实例化一个网站应用实例
?>
```

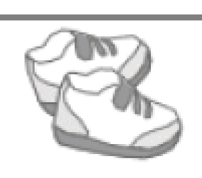
（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

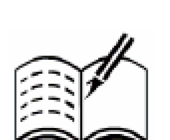
（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                       //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                      //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");      //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
              if(isset($_POST['user']) && isset($_POST['pass'])){
                    $db = M();                  //实例化模型类，参数数据表名称，不包含前缀
                    $select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                          //执行查询语句，验证用户名和密码是否正确
                    if($select){
                          $_SESSION['admin']=$_POST['user'];  //将登录用户名存储到 SESSION 中
                          $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');//页面
重定向
                    }else{
                          $this->redirect('Index/index','', 2,'用户名或者密码不正确！'); //页面重定向
                    }
              }else{
                    $this->redirect('Index/index','', 2,'用户名、密码不能为空！');      //页面重定向
              }
```





（2）在 002 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');            //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                      //定义项目名称
define('APP_PATH', './App');                    //定义项目路径
require(THINK_PATH."/ThinkPHP.php");            //加载框架入口文件
App::run();                                     //实例化一个网站应用实例
?>
```

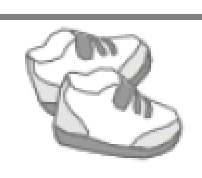
（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

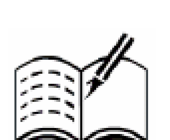
（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                         //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                    //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");    //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
            if(isset($_POST['user']) && isset($_POST['pass'])){
                $db = M();                  //实例化模型类，参数数据表名称，不包含前缀
                $select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                    //执行查询语句，验证用户名和密码是否正确
                if($select){
                    $_SESSION['admin']=$_POST['user'];  //将登录用户名存储到 SESSION 中
                    $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');//页面
重定向
                }else{
                    $this->redirect('Index/index','', 2,'用户名或者密码不正确！'); //页面重定向
                }
            }else{
                $this->redirect('Index/index','', 2,'用户名、密码不能为空！');      //页面重定向
            }
```





Note

（2）在 002 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');            //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                      //定义项目名称
define('APP_PATH', './App');                    //定义项目路径
require(THINK_PATH."/ThinkPHP.php");            //加载框架入口文件
App::run();                                     //实例化一个网站应用实例
?>
```

（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，编辑 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                         //关闭调试模式
);
?>
```

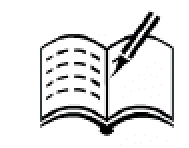
（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                        //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");        //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
              if(isset($_POST['user']) && isset($_POST['pass'])){
                    $db = M();                  //实例化模型类，参数数据表名称，不包含前缀
                    $select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                    //执行查询语句，验证用户名和密码是否正确
                    if($select){
                          $_SESSION['admin']=$_POST['user'];  //将登录用户名存储到 SESSION 中
                          $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');//页面
重定向
                    }else{
                          $this->redirect('Index/index','', 2,'用户名或者密码不正确！'); //页面重定向
                    }
              }else{
                    $this->redirect('Index/index','', 2,'用户名、密码不能为空！');       //页面重定向
              }
```



```
        }
        $this->display();
    }
```

定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。其代码如下：



```
    public function main(){
        $db = M('User');                //实例化模型类，参数数据表名称，不包含前缀
        //进行分页数据查询，注意 page 方法的参数的前面部分是当前的页数使用 $_GET[p]获取
        if(isset($_GET['p'])){          //判断分页变量是否存在
            $p=$_GET['p'];
        }else{
            $p=1;
        }
        $list = $db->where('address='."'长春市'")->order('id desc')->page($p.',1')->select();//查询数据
        $this->assign('select',$list);         //赋值数据集
        import("ORG.Util.Page");               //导入分页类
        $count = $db->where('address='."'长春市'")->count();    //查询满足要求的总记录数
        $Page = new Page($count,1);        //实例化分页类，传入总记录数和每页显示的记录数
        $show = $Page->show();             //分页显示输出
        $this->assign('page',$show);       //赋值分页输出
        $this->display(main);              //输出模板
    }
```

定义 validatorcode()方法，应用 GD 库中的函数，根据超链接传递的值生成用户登录的验证码。其代码如下：

```
    public function validatorcode(){
      header('content-type:image/png');                             //定义标题 png 格式图像
      $im = imagecreate(65, 25);                                    //定义画布
      imagefill($im, 0, 0, imagecolorallocate($im, 200, 200, 200));   //区域填充
      $validatorCode = $_GET['code'];                               //获取提交的值
      imagestring($im, rand(3, 5), 10, 3, substr($validatorCode, 0, 1), imagecolorallocate($im, 0, rand(0,
255), rand(0, 255)));
      imagestring($im, rand(3, 5), 25, 6, substr($validatorCode, 1, 1), imagecolorallocate($im, rand(0,
255), 0, rand(0, 255)));
      imagestring($im, rand(3, 5), 36, 9, substr($validatorCode, 2, 1), imagecolorallocate($im, rand(0,
255), rand(0, 255), 0));
      imagestring($im, rand(3, 5), 48, 12, substr($validatorCode, 3, 1), imagecolorallocate($im, 0, rand(0,
255), rand(0, 255)));
      imagepng($im);                                                //生成 PNG 图像
      imagedestroy();                                               //销毁图像
    }
```

（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作。其关键代码如下：

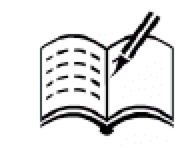
```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<js href="__ROOT__/Public/Js/check.js" />
```



```
        }
        $this->display();
    }
```

定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。其代码如下：

```
    public function main(){
        $db = M('User');                //实例化模型类，参数数据表名称，不包含前缀
        //进行分页数据查询，注意 page 方法的参数的前面部分是当前的页数使用 $_GET[p]获取
        if(isset($_GET['p'])){          //判断分页变量是否存在
            $p=$_GET['p'];
        }else{
            $p=1;
        }
        $list = $db->where('address='."'长春市'")->order('id desc')->page($p.',1')->select();//查询数据
        $this->assign('select',$list);          //赋值数据集
        import("ORG.Util.Page");                //导入分页类
        $count = $db->where('address='."'长春市'")->count();     //查询满足要求的总记录数
        $Page = new Page($count,1);             //实例化分页类，传入总记录数和每页显示的记录数
        $show = $Page->show();                  //分页显示输出
        $this->assign('page',$show);            //赋值分页输出
        $this->display(main);                   //输出模板
    }
```

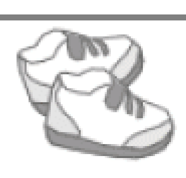


定义 validatorcode()方法，应用 GD 库中的函数，根据超链接传递的值生成用户登录的验证码。其代码如下：

```
    public function validatorcode(){
        header('content-type:image/png');                               //定义标题 png 格式图像
        $im = imagecreate(65, 25);                                      //定义画布
        imagefill($im, 0, 0, imagecolorallocate($im, 200, 200, 200));   //区域填充
        $validatorCode = $_GET['code'];                                 //获取提交的值
        imagestring($im, rand(3, 5), 10, 3, substr($validatorCode, 0, 1), imagecolorallocate($im, 0, rand(0, 255), rand(0, 255)));
        imagestring($im, rand(3, 5), 25, 6, substr($validatorCode, 1, 1), imagecolorallocate($im, rand(0, 255), 0, rand(0, 255)));
        imagestring($im, rand(3, 5), 36, 9, substr($validatorCode, 2, 1), imagecolorallocate($im, rand(0, 255), rand(0, 255), 0));
        imagestring($im, rand(3, 5), 48, 12, substr($validatorCode, 3, 1), imagecolorallocate($im, 0, rand(0, 255), rand(0, 255)));
        imagepng($im);                                                  //生成 PNG 图像
        imagedestroy();                                                 //销毁图像
    }
```

（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作。其关键代码如下：

```
    <link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
    <js href="__ROOT__/Public/Js/check.js" />
```

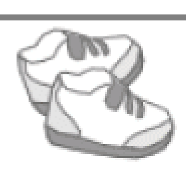




```
<form name="form1" method="post"   action="__URL__/index" onSubmit="return chkinput(this)" >
<table width="265" border="0" cellspacing="0" cellpadding="0">
 <tr>
      <td class="title" id="td">用户名：</td>
      <td><input name="user" type="text" size="15" /></td>
 </tr>
 <tr>
      <td class="title" id="td">密码：</td>
      <td><input name="pass" type="password" size="15" /></td>
 </tr>
 <tr>
      <td class="title" id="td">验证码：</td>
      <td>
           <input type="text" name="validatorCode" size="10" />
           <input type="hidden" name="defValidatorCode" value="" />
<script language="javascript">
 var num1=Math.round(Math.random()*10000000);              //生成随机数
     var num=num1.toString().substr(0,4);                  //截取随机数的前 4 个字符
     document.write("<img  name=codeimg  src='__URL__/validatorcode?code="+num+"'>"); //将截取值传递到图像处理页中
     form1.defValidatorCode.value=num;                     //将截取值赋给表单中的隐藏域
 function reCode(){                                        //定义方法，重新生成验证码
      var num1=Math.round(Math.random()*10000000);         //生成随机数
      var num=num1.toString().substr(0,4);                 //截取随机数
      document.codeimg.src="__URL__/validatorcode?code="+num;//将截取值传递到图像处理页中
           form1.defValidatorCode.value=num;               //将截取值赋给表单中的隐藏域
    }
</script>
           <a href="javascript:reCode()" class="content">看不清</a>
           </td>
 </tr>
</table>
 <input type="image" name="imageField" id="imageField" src="__ROOT__/Public/images/66_05.gif" />
</form>
```

（7）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 SESSION 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
<table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor="#FFFFFF">
      <tr>
        <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">当前登录用户：{$Think.session.admin}</td>
      </tr>
      <foreach name='select' item='user' >
      <tr class="content">
        <td bgcolor="#FFFFFF"> {$user.id}</td>
        <td bgcolor="#FFFFFF"> {$user.user}</td>
        <td bgcolor="#FFFFFF"> {$user.address}</td>
```

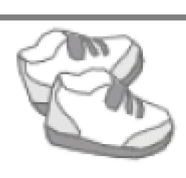





```
<form name="form1" method="post"  action="__URL__/index" onSubmit="return chkinput(this)" >
<table width="265" border="0" cellspacing="0" cellpadding="0">
 <tr>
     <td class="title" id="td">用户名：</td>
     <td><input name="user" type="text" size="15" /></td>
 </tr>
 <tr>
     <td class="title" id="td">密码：</td>
     <td><input name="pass" type="password" size="15" /></td>
 </tr>
 <tr>
     <td class="title" id="td">验证码：</td>
     <td>
         <input type="text" name="validatorCode" size="10" />
         <input type="hidden" name="defValidatorCode" value="" />
<script language="javascript">
 var num1=Math.round(Math.random()*10000000);              //生成随机数
     var num=num1.toString().substr(0,4);                  //截取随机数的前 4 个字符
     document.write("<img  name=codeimg  src='__URL__/validatorcode?code="+num+"'>"); //将截
取值传递到图像处理页中
     form1.defValidatorCode.value=num;                     //将截取值赋给表单中的隐藏域
 function reCode(){                                        //定义方法，重新生成验证码
     var num1=Math.round(Math.random()*10000000);          //生成随机数
     var num=num1.toString().substr(0,4);                  //截取随机数
     document.codeimg.src="__URL__/validatorcode?code="+num;//将截取值传递到图像处理页中
         form1.defValidatorCode.value=num;                 //将截取值赋给表单中的隐藏域
   }
</script>
         <a href="javascript:reCode()" class="content">看不清</a>
         </td>
 </tr>
</table>
 <input type="image" name="imageField" id="imageField" src="__ROOT__/Public/images/66_05.gif" />
</form>
```

（7）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 SESSION 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
<table  width="405"  border="1"  cellpadding="1"  cellspacing="1"  bgcolor="#99CC33"  bordercolor=
"#FFFFFF">
     <tr>
       <td  colspan="3"  bgcolor="#FFFFFF"  class="title"  align="center">当前登录用户：{$Think.session.
admin}</td>
     </tr>
     <foreach name='select' item='user' >
     <tr class="content">
       <td bgcolor="#FFFFFF"> {$user.id}</td>
       <td bgcolor="#FFFFFF"> {$user.user}</td>
       <td bgcolor="#FFFFFF"> {$user.address}</td>
```

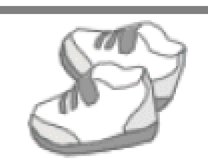





```
<form name="form1" method="post"  action="__URL__/index" onSubmit="return chkinput(this)" >
 <table width="265" border="0" cellspacing="0" cellpadding="0">
  <tr>
      <td class="title" id="td">用户名：</td>
      <td><input name="user" type="text" size="15" /></td>
  </tr>
  <tr>
      <td class="title" id="td">密码：</td>
      <td><input name="pass" type="password" size="15" /></td>
  </tr>
  <tr>
      <td class="title" id="td">验证码：</td>
      <td>
          <input type="text" name="validatorCode" size="10" />
          <input type="hidden" name="defValidatorCode" value="" />
 <script language="javascript">
  var num1=Math.round(Math.random()*10000000);              //生成随机数
     var num=num1.toString().substr(0,4);                   //截取随机数的前 4 个字符
     document.write("<img  name=codeimg  src='__URL__/validatorcode?code="+num+"'>"); //将截
取值传递到图像处理页中
     form1.defValidatorCode.value=num;                      //将截取值赋给表单中的隐藏域
  function reCode(){                                        //定义方法，重新生成验证码
      var num1=Math.round(Math.random()*10000000);          //生成随机数
      var num=num1.toString().substr(0,4);                  //截取随机数
      document.codeimg.src="__URL__/validatorcode?code="+num;//将截取值传递到图像处理页中
          form1.defValidatorCode.value=num;                 //将截取值赋给表单中的隐藏域
     }
 </script>
          <a href="javascript:reCode()" class="content">看不清</a>
          </td>
  </tr>
 </table>
  <input type="image" name="imageField" id="imageField" src="__ROOT__/Public/images/66_05.gif" />
 </form>
```

（7）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 SESSION 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
 <table  width="405"  border="1"  cellpadding="1"  cellspacing="1"  bgcolor="#99CC33"  bordercolor=
"#FFFFFF">
   <tr>
     <td  colspan="3"  bgcolor="#FFFFFF"  class="title"  align="center">当前登录用户：{$Think.session.
admin}</td>
   </tr>
   <foreach name='select' item='user' >
   <tr class="content">
     <td bgcolor="#FFFFFF"> {$user.id}</td>
     <td bgcolor="#FFFFFF"> {$user.user}</td>
     <td bgcolor="#FFFFFF"> {$user.address}</td>
```



```
            </tr>
            </foreach>
              <tr class="content">
                <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
                </tr>
        </table>
```



技术要点

本实例中的技术要点可以归纳为两个方面：第一个方面是通过 ThinkPHP 框架中的分页类以及连贯操作 Page()方法实现数据的分页输出；第二个方面是应用{$Think.session.session_id|md5 }标签输出 SESSION 变量。

（1）通过 import("ORG.Util.Page");导入分页类。

```
import("ORG.Util.Page");                    //导入分页类
```

（2）实例化分页类，同时可以传递三个参数：第一个是页面总记录数，第二个是每页显示的记录数，第三个为可选参数，通过分页超链接的值。

```
$count = $db->where('address='."'长春市'")->count();       //查询满足要求的总记录数
$Page = new Page($count,1);                  //实例化分页类，传入总记录数和每页显示的记录数
```

（3）调用分页类中的 show()方法，输出分页超链接。

```
$show = $Page->show();                       //分页显示输出
```

（4）Page()方法：查询分页。属于新增特性，可以更加快速地进行分页查询。Page 方法的用法和 limit 方法类似，格式为：

```
Page('page[,listRows]')
```

参数 page 表示当前的页数，参数 listRows 表示每页显示的记录数。例如，Page('2,10')。表示每页显示 10 条记录，获取第 2 页的数据

listRow 如果不写的话，会读取 limit('length') 的值，例如，limit(25)->page(3);。表示每页显示 25 条记录，获取第 3 页的数据。如果 limit 也没有设置的话，则默认为每页显示 20 条记录。

指点迷津：

在通过 Page()方法进行数据的分页查询时，page 方法的第一个参数是当前的页数，需要使用$_GET['p']，第二个参数是当前页显示的记录数。

实例 255　通过验证码类和分页类完成用户登录和分页输出

（实例位置：配套资源\SL\15\255　视频位置：配套资源\SP\15\255）

实例说明

本实例仍然以用户登录和数据的输出为背景，应用 ThinkPHP 中提供的验证码类和分



页类生成验证码，完成数据的分页输出。其运行效果如图 15.3 所示。



图 15.3　通过验证码类和分页类完成用户登录和分页输出

实现过程

在讲解实例的实现步骤过程中，省略了项目目录的创建、入口文件的编写和配置文件的设置，其具体步骤可以参考实例 253 或者实例 254。这里讲解在控制器中如何完成数据的添加和查询操作。

（1）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证 SESSION 变量存储的验证码与用户提交的验证码是否相同，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();
header("Content-Type:text/html; charset=utf-8");            //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
              if(isset($_POST['user']) && isset($_POST['pass']) && isset($_POST['validatorCode'])){
                    if($_SESSION['verify'] == md5($_POST['validatorCode'])) {//验证验证码是否正确
                          $db = M();              //实例化模型类，参数数据表名称，不包含前缀
                          $select = $db->query("select * from think_user where user='".$_POST['user']."' and pass='".$_POST['pass']."'");                //执行查询语句，验证用户名和密码是否正确
                          if($select){
                                $_SESSION['admin']=$_POST['user'];
                                $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！'); //页面重定向
                          }else{
                                $this->redirect('Index/index','', 2,'用户名或者密码不正确！');//页面重定向
                          }
                    }else{
                          $this->redirect('Index/index','', 2,'验证码不正确！');      //页面重定向
                    }
              }else{
```



页类生成验证码，完成数据的分页输出。其运行效果如图 15.3 所示。



图 15.3　通过验证码类和分页类完成用户登录和分页输出

实现过程

在讲解实例的实现步骤过程中，省略了项目目录的创建、入口文件的编写和配置文件的设置，其具体步骤可以参考实例 253 或者实例 254。这里讲解在控制器中如何完成数据的添加和查询操作。

（1）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证 SESSION 变量存储的验证码与用户提交的验证码是否相同，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 SESSION 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();
header("Content-Type:text/html; charset=utf-8");            //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['user'])){
              if(isset($_POST['user']) && isset($_POST['pass']) && isset($_POST['validatorCode'])){
                    if($_SESSION['verify'] == md5($_POST['validatorCode'])) {//验证验证码是否正确
                          $db = M();              //实例化模型类，参数数据表名称，不包含前缀
                          $select = $db->query("select * from think_user where user='".$_POST['user']."' and pass='".$_POST['pass']."'");               //执行查询语句，验证用户名和密码是否正确
                          if($select){
                                $_SESSION['admin']=$_POST['user'];
                                $this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！'); //页面重定向
                          }else{
                                $this->redirect('Index/index','', 2,'用户名或者密码不正确！');//页面重定向
                          }
                    }else{
                          $this->redirect('Index/index','', 2,'验证码不正确！');       //页面重定向
                    }
              }else{
```



```
                $this->redirect('Index/index','', 2,'用户名、密码不能为空！'); //页面重定向
            }
        }
        $this->display();
    }
```



定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。这里应用的是 Page 类和 Limit()方法完成数据的分页输出，其代码如下：

```
    public function main(){
        $db = M('User');                    //实例化模型类，参数数据表名称，不包含前缀
        import("ORG.Util.Page");                             //导入分页类
        $count = $db->count();                               //统计总记录数
        //$count = $User->where("status=1")->count();        //查询满足要求的总记录数
        $Page = new Page($count,1);         //实例化分页类 传入总记录数和每页显示的记录数
        $show = $Page->show();                               //分页显示输出
        //进行分页数据查询，注意 limit 方法的参数要使用 Page 类的属性
        $list = $db->order('id')->limit($Page->firstRow.','.$Page->listRows)->select();
        $this->assign('select',$list);                       //赋值数据集
        $this->assign('page',$show);                         //赋值分页输出
        $this->display(main);                                //输出模板
    }
```

定义 verify()方法，载入 ThinkPHP 中提供的验证码扩展类，调用 buildImageVerify()方法生成验证码。其代码如下：

```
    public function verify(){
      import("ORG.Util.Image");                              //载入验证码类
      image::buildImageVerify(4,5,'png','65','30','verify');  //生成验证码
    }
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作，通过 img 标签输出生成的验证码。其关键代码如下：

```
    <link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
    <js href="__ROOT__/Public/Js/check.js" />
    <form name="form1" method="post"   action="__URL__/index" onSubmit="return chkinput(this)" >
    <table width="265" border="0" cellspacing="0" cellpadding="0">
      <tr>
          <td class="title" id="td">用户名：</td>
          <td><input name="user" type="text" size="15" /></td>
      </tr>
      <tr>
          <td class="title" id="td">密码：</td>
          <td><input name="pass" type="password" size="15" /></td>
      </tr>
      <tr>
          <td class="title" id="td">验证码：</td>
          <td><input type="text" name="validatorCode" size="10" /></td>
```



```
                $this->redirect('Index/index','', 2,'用户名、密码不能为空！'); //页面重定向
            }
        }
        $this->display();
    }
```



定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。这里应用的是 Page 类和 Limit()方法完成数据的分页输出，其代码如下：

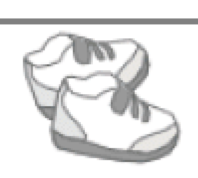
```
    public function main(){
        $db = M('User');                    //实例化模型类，参数数据表名称，不包含前缀
        import("ORG.Util.Page");                        //导入分页类
        $count = $db->count();                          //统计总记录数
        //$count = $User->where("status=1")->count();   //查询满足要求的总记录数
        $Page = new Page($count,1);         //实例化分页类 传入总记录数和每页显示的记录数
        $show = $Page->show();                          //分页显示输出
        //进行分页数据查询，注意 limit 方法的参数要使用 Page 类的属性
        $list = $db->order('id')->limit($Page->firstRow.','.$Page->listRows)->select();
        $this->assign('select',$list);                  //赋值数据集
        $this->assign('page',$show);                    //赋值分页输出
        $this->display(main);                           //输出模板
    }
```

定义 verify()方法，载入 ThinkPHP 中提供的验证码扩展类，调用 buildImageVerify()方法生成验证码。其代码如下：

```
    public function verify(){
      import("ORG.Util.Image");                         //载入验证码类
      image::buildImageVerify(4,5,'png','65','30','verify');   //生成验证码
    }
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作，通过 img 标签输出生成的验证码。其关键代码如下：

```
    <link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
    <js href="__ROOT__/Public/Js/check.js" />
    <form name="form1" method="post"   action="__URL__/index" onSubmit="return chkinput(this)" >
    <table width="265" border="0" cellspacing="0" cellpadding="0">
      <tr>
            <td class="title" id="td">用户名：</td>
            <td><input name="user" type="text" size="15" /></td>
      </tr>
      <tr>
            <td class="title" id="td">密码：</td>
            <td><input name="pass" type="password" size="15" /></td>
      </tr>
      <tr>
            <td class="title" id="td">验证码：</td>
            <td><input type="text" name="validatorCode" size="10" /></td>
```



```
            <td><img src="__APP__/Index/verify/" /></td>
        </tr>
    </table>
    </form>
```



Note

脚下留神：

这里应用的验证码是区分字母的大小写的。

（3）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 SESSION 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
    <table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor=
"#FFFFFF">
        <tr>
          <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">当前登录用户：{$Think.session.
admin}</td>
        </tr>
        <foreach name='select' item='user' >
        <tr class="content">
          <td bgcolor="#FFFFFF"> {$user.id}</td>
          <td bgcolor="#FFFFFF"> {$user.user}</td>
          <td bgcolor="#FFFFFF"> {$user.address}</td>
        </tr>
        </foreach>
          <tr class="content">
          <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
          </tr>
    </table>
```

技术要点

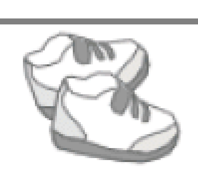
buildImageVerify()方法的语法如下：

```
buildImageVerify($length,$mode,$type,$width,$height,$verifyName)
```

参数说明如表 15.1 所示。

表 15.1　验证码类中 buildImageVerify()方法的参数说明

参　数	说　明
length	验证码的长度，默认为 4 位数
mode	验证字符串的类型，默认为数字，其他支持类型有 0 字母 1 数字 2 大写字母 3 小写字母 4 中文 5 混合（去掉了容易混淆的字符 oOLl 和数字 01）
type	验证码的图片类型，默认为 png
width	验证码的宽度，默认会自动根据验证码长度自动计算
height	验证码的高度，默认为 22
verifyName	验证码的 SESSION 记录名称，默认为 verify



```
        <td><img src="__APP__/Index/verify/" /></td>
    </tr>
  </table>
  </form>
```



Note

脚下留神：

这里应用的验证码是区分字母的大小写的。

（3）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 SESSION 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
    <table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor=
"#FFFFFF">
      <tr>
        <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">当前登录用户：{$Think.session.
admin}</td>
      </tr>
      <foreach name='select' item='user' >
      <tr class="content">
        <td bgcolor="#FFFFFF"> {$user.id}</td>
        <td bgcolor="#FFFFFF"> {$user.user}</td>
        <td bgcolor="#FFFFFF"> {$user.address}</td>
      </tr>
      </foreach>
       <tr class="content">
        <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
        </tr>
    </table>
```

技术要点

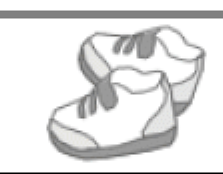
buildImageVerify()方法的语法如下：

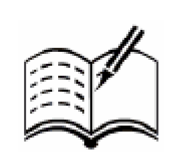
```
buildImageVerify($length,$mode,$type,$width,$height,$verifyName)
```

参数说明如表 15.1 所示。

表 15.1　验证码类中 buildImageVerify()方法的参数说明

参　　数	说　　明
length	验证码的长度，默认为 4 位数
mode	验证字符串的类型，默认为数字，其他支持类型有 0 字母 1 数字 2 大写字母 3 小写字母 4 中文 5 混合（去掉了容易混淆的字符 oOLl 和数字 01）
type	验证码的图片类型，默认为 png
width	验证码的宽度，默认会自动根据验证码长度自动计算
height	验证码的高度，默认为 22
verifyName	验证码的 SESSION 记录名称，默认为 verify





多学两招：

生成验证码之后，需要在模板页中通过输出生成的验证码图像；在控制器中通过如下代码：

```
if($_SESSION['verify'] != md5($_POST['verify'])) {
    $this->error('验证码错误！');
}
```

验证用户提交的验证码是否正确。

实例 256 通过 ThinkPHP 中的扩展类生成中文验证码

（实例位置：配套资源\SL\15\256 视频位置：配套资源\SP\15\256）

实例说明

通过 ThinkPHP 框架实现用户登录功能，载入 ThinkPHP\Lib\ORG\Util\Image.class.php 中的 Image 类，调用其中的 GBVerify()方法，生成中文验证码。其运行效果如图 15.4 所示。



图 15.4 通过 ThinkPHP 中的扩展类生成中文验证码

实现过程

本实例的实现过程与实例 255 相同，都是实现用户登录和数据的分页输出功能，所以有关本实例的实现过程可以参考实例 255，这里不再赘述。这里将重点讲解它们的不同之处，用户登录中采用的中文验证码。

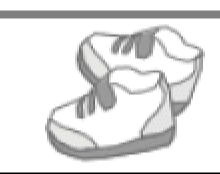
定位到\App\Lib\Action\目录下，在项目控制器的 Index 模块中，定义 verify()方法，载入 ThinkPHP 中提供的验证码扩展类，调用 GBVerify()方法生成中文验证码。其代码如下：

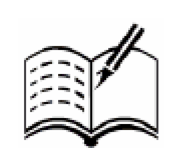
```
public function verify(){
  import("ORG.Util.Image");                                //载入验证码类
  image::buildImageVerify(4,5,'png','65','30','verify');   //生成中文验证码
}
```

这是本实例中唯一与上一实例的不同之处，也是本实例的一个亮点，有关本实例其他步骤的实现过程可以参考上一实例或者配套资源源码，这里不再赘述。

指点迷津：

要应用 ThinkPHP 2.1 版本提供的 Image 扩展类完成中文验证码的生成，需要对\Lib\ORG\Util\Image.class.php 文件中 386 行的“msubstr”进行修改，将其修改为“String::msubstr”，否则直接运行程序是不能输出中文验证码的。





多学两招：

生成验证码之后，需要在模板页中通过输出生成的验证码图像；在控制器中通过如下代码：

```
if($_SESSION['verify'] != md5($_POST['verify'])) {
    $this->error('验证码错误！');
}
```

验证用户提交的验证码是否正确。

实例 256　通过 ThinkPHP 中的扩展类生成中文验证码

（实例位置：配套资源\SL\15\256　视频位置：配套资源\SP\15\256）

实例说明

通过 ThinkPHP 框架实现用户登录功能，载入 ThinkPHP\Lib\ORG\Util\Image.class.php 中的 Image 类，调用其中的 GBVerify()方法，生成中文验证码。其运行效果如图 15.4 所示。



图 15.4　通过 ThinkPHP 中的扩展类生成中文验证码

实现过程

本实例的实现过程与实例 255 相同，都是实现用户登录和数据的分页输出功能，所以有关本实例的实现过程可以参考实例 255，这里不再赘述。这里将重点讲解它们的不同之处，用户登录中采用的中文验证码。

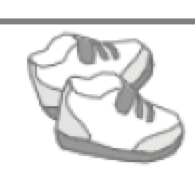
定位到\App\Lib\Action\目录下，在项目控制器的 Index 模块中，定义 verify()方法，载入 ThinkPHP 中提供的验证码扩展类，调用 GBVerify()方法生成中文验证码。其代码如下：

```
public function verify(){
  import("ORG.Util.Image");                              //载入验证码类
  image::buildImageVerify(4,5,'png','65','30','verify');   //生成中文验证码
}
```

这是本实例中唯一与上一实例的不同之处，也是本实例的一个亮点，有关本实例其他步骤的实现过程可以参考上一实例或者配套资源源码，这里不再赘述。

指点迷津：

要应用 ThinkPHP 2.1 版本提供的 Image 扩展类完成中文验证码的生成，需要对\Lib\ORG\Util\Image.class.php 文件中 386 行的“msubstr”进行修改，将其修改为“String::msubstr”，否则直接运行程序是不能输出中文验证码的。



技术要点

GBVerify ()方法的语法如下：



```
GBVerify ($length,$type,$width,$height,$fontface,$verifyName)
```

参数说明如表 15.2 所示。

表 15.2　验证码类中 GBVerify()方法的参数说明

参　　数	说　　明
length	验证码的长度，默认为 4 位数
type	验证码的图片类型，默认为 png
type	验证码的图片类型，默认为 png
width	验证码的宽度，默认会自动根据验证码长度自动计算
height	验证码的高度，默认为 50
verifyName	验证码的 SESSION 记录名称，默认为 verify
fontface	使用的字体，使用完整文件名或者放到图像类所在的目录下面，默认使用的字体文件是 simhei.tt
verifyName	验证码的 SESSION 记录名称，默认为 verify

指点迷津：

在应用 GBVerify ()方法生成中文验证码时，必须注意以下几个问题。

（1）PHP 是否已经安装 GD 库支持;

（2）输出之前是否有任何的输出（尤其是 UTF-8 的头信息输出）;

（3）Image 类库是否正确导入;

（4）如果是中文验证码检查是否有复制字体文件到类库所在目录。

实例 257　可以传递查询条件的分页

（实例位置：配套资源\SL\15\257　视频位置：配套资源\SP\15\257）

实例说明

本实例通过 TinkPHP 框架开发一个站内搜索的功能，并且对查询结果进行分页输出。其运行结果如图 15.5 所示。

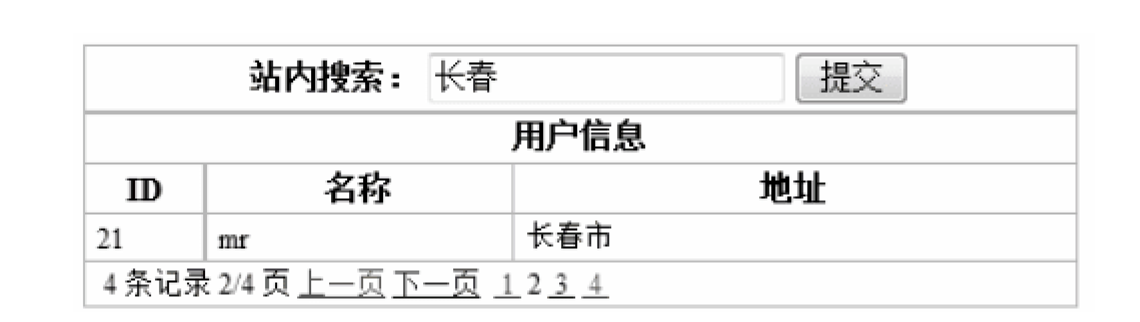


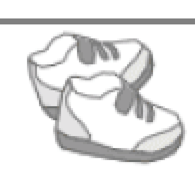
图 15.5　可以传递查询条件的分页

实现过程

具体步骤如下：

（1）创建 005 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。

（2）在 005 项目根目录下，编辑 index.php 入口文件。其关键代码如下：



技术要点

GBVerify ()方法的语法如下：

```
GBVerify ($length,$type,$width,$height,$fontface,$verifyName)
```

参数说明如表 15.2 所示。


表 15.2　验证码类中 GBVerify()方法的参数说明

参　　数	说　　明
length	验证码的长度，默认为 4 位数
type	验证码的图片类型，默认为 png
type	验证码的图片类型，默认为 png
width	验证码的宽度，默认会自动根据验证码长度自动计算
height	验证码的高度，默认为 50
verifyName	验证码的 SESSION 记录名称，默认为 verify
fontface	使用的字体，使用完整文件名或者放到图像类所在的目录下面，默认使用的字体文件是 simhei.tt
verifyName	验证码的 SESSION 记录名称，默认为 verify

指点迷津：

在应用 GBVerify ()方法生成中文验证码时，必须注意以下几个问题。

（1）PHP 是否已经安装 GD 库支持;

（2）输出之前是否有任何的输出（尤其是 UTF-8 的头信息输出）;

（3）Image 类库是否正确导入;

（4）如果是中文验证码检查是否有复制字体文件到类库所在目录。

实例 257　可以传递查询条件的分页

（实例位置：配套资源\SL\15\257　视频位置：配套资源\SP\15\257）

实例说明

本实例通过 TinkPHP 框架开发一个站内搜索的功能，并且对查询结果进行分页输出。其运行结果如图 15.5 所示。

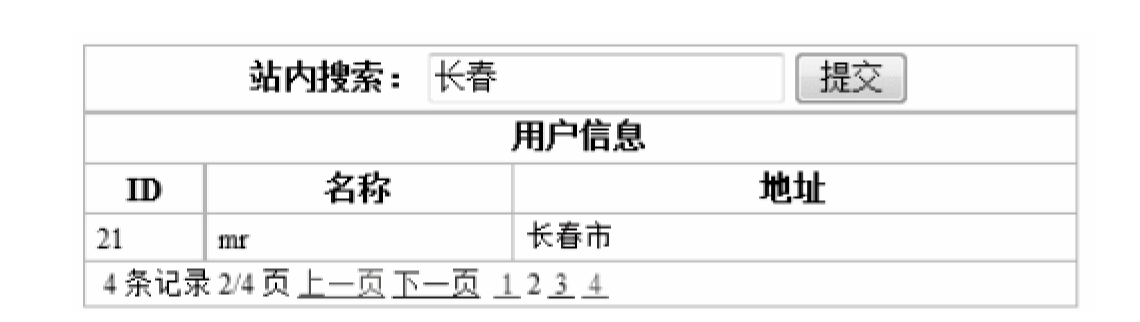


图 15.5　可以传递查询条件的分页

实现过程

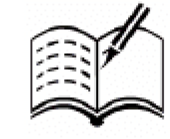
具体步骤如下：

（1）创建 005 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。

（2）在 005 项目根目录下，编辑 index.php 入口文件。其关键代码如下：



```
<?php
define('THINK_PATH', '../ThinkPHP');          //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                    //定义项目名称
define('APP_PATH', './App');                  //定义项目路径
require(THINK_PATH."/ThinkPHP.php");          //加载框架入口文件
App::run();                                   //实例化一个网站应用实例
?>
```



（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，创建 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                  //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法。首先，获取查询关键字（通过$_POST 获取表单提交的查询关键字；通过$_GET 获取超链接传递的查询关键字，这是查询结果分页输出的关键），并且将其赋给指定的变量。其代码如下：

```
<?php
session_start();                                        //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");        //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['key'])){                       //判断查询的关键字是否存在
              $key=$_POST['key'];
        }else if(isset($_GET['key'])){
              $key=$_GET['key'];
        }
```

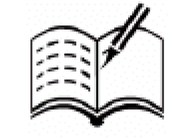
然后，实例化基础模型类，载入分页类。其代码如下：

```
$db = M('User');                    //实例化模型类，参数数据表名称，不包含前缀
import("ORG.Util.Page");            //导入分页类
```

接着，判断查询关键字是否存在，如果存在，则根据获取的关键字执行模糊查询，并且对查询结果进行分页输出；如果不存在，则对数据库中的所有数据进行分页输出。其代



```
<?php
define('THINK_PATH', '../ThinkPHP');          //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                    //定义项目名称
define('APP_PATH', './App');                  //定义项目路径
require(THINK_PATH."/ThinkPHP.php");          //加载框架入口文件
App::run();                                   //实例化一个网站应用实例
?>
```



（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，创建 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
  'DB_TYPE'=> 'pdo',
  //注意 DSN 的配置针对不同的数据库有所区别
  'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
  'DB_USER'=>'root',
  'DB_PWD'=>'111',
  'DB_PREFIX'=>'think_',
  //其他项目配置参数……
  'APP_DEBUG' => false,                 //关闭调试模式
);
?>
```

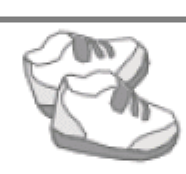
（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法。首先，获取查询关键字（通过$_POST 获取表单提交的查询关键字；通过$_GET 获取超链接传递的查询关键字，这是查询结果分页输出的关键），并且将其赋给指定的变量。其代码如下：

```
<?php
session_start();                                        //初始化 SESSION 变量
header("Content-Type:text/html; charset=utf-8");        //设置页面编码格式
class IndexAction extends Action{
  public function index(){
        if(isset($_POST['key'])){                       //判断查询的关键字是否存在
              $key=$_POST['key'];
        }else if(isset($_GET['key'])){
              $key=$_GET['key'];
        }
```

然后，实例化基础模型类，载入分页类。其代码如下：

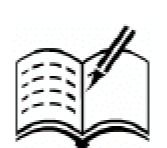
```
$db = M('User');                        //实例化模型类，参数数据表名称，不包含前缀
import("ORG.Util.Page");                //导入分页类
```

接着，判断查询关键字是否存在，如果存在，则根据获取的关键字执行模糊查询，并且对查询结果进行分页输出；如果不存在，则对数据库中的所有数据进行分页输出。其代码



码如下：

```
if($key!=''){
    $map="user like('%".$key."%') or address like('%".$key."%') ";
    $list = $db->where($map)->order('id desc')->page($_GET['p'].',1')->select();
    $this->assign('select',$list);           //赋值数据集
    $count = $db->where($map)->count();      //查询满足要求的总记录数
    $Page = new Page($count,1,'key='.$key);  //实例化分页类 传入总记录数、每页显示的记录数和查询的关键字
    $show = $Page->show();                    //分页显示输出
    $this->assign('page',$show);              //赋值分页输出
}else{
    $count = $db->count();                    //统计总记录数
    $Page = new Page($count,1); //实例化分页类，传入总记录数和每页显示的记录数
    $show = $Page->show();                    //分页显示输出
    //进行分页数据查询，注意 limit 方法的参数要使用 Page 类的属性
    $list = $db->order('id')->limit($Page->firstRow.','.$Page->listRows)->select();
    $this->assign('select',$list);            //赋值数据集
    $this->assign('page',$show);              //赋值分页输出
}
```



最后，应用 display()方法指定模板页。其代码如下：

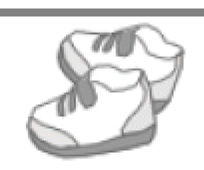
```
        $this->display();                     //输出模板
    }
}
?>
```

（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件；创建表单，提交查询的关键字；应用 foreach 语句完成查询结果的分页输出；应用{$page}输出分页超链接。其关键代码如下：

```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<form id="form1" name="form1" method="post" action="__URL__/index">
    站内搜索：
    <input type="text" name="key" id="key" />
    <input type="submit" name="button" id="button" value="提交" />
</form>
<foreach name='select' item='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
</foreach>
<tr class="content">
    <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
</tr>
```

技术要点

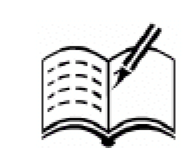
查询条件在分页超链接中的传递，关键是在实例化 Page 类中第三个参数（$parameter）



的设置，此参数定义的是在分页超链接中传递的参数名称和对应的参数值。其关键代码如下：

```
$Page = new Page($count,1,'key='.$key);
```

第一个参数是查询的总记录数；第二个参数是每页显示的记录数；第三参数是在分页超链接中传递的参数和参数值。



实例 258　应用 ThinkPHP 中的扩展类上传文件

（实例位置：配套资源\SL\15\258　视频位置：配套资源\SP\15\258）

实例说明

本实例应用 ThinkPHP 中的文件上传扩展类（Lib\ORG\Net\UploadFile.class.php）实现文件上传的功能，上传 jpg、gif 或者 png 格式的图片到服务器指定的文件夹下，文件上传和浏览的运行效果如图 15.6 所示。



图 15.6　应用 ThinkPHP 中的扩展类上传文件

实现过程

具体步骤如下：

（1）创建 006 项目根目录，在根目录下创建项目文件夹 App 和 Public 文件夹存储 CSS、图片和 JS 脚本等文件。

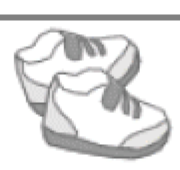
（2）在 006 项目根目录下，创建 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP');          //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'App');                    //定义项目名称
define('APP_PATH', './App');                  //定义项目路径
require(THINK_PATH."/ThinkPHP.php");          //加载框架入口文件
App::run();                                   //实例化一个网站应用实例
?>
```

（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，创建 config.php 文件，通过 PDO 连接 MySQL 数据库。其代码如下：

```
<?php
return array(
```



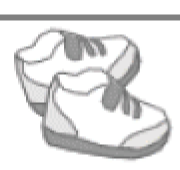

```
    'DB_TYPE'=> 'pdo',
    // 注意 DSN 的配置针对不同的数据库有所区别
    'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
    'DB_USER'=>'root',
    'DB_PWD'=>'111',
    'DB_PREFIX'=>'think_',
    // 其他项目配置参数……
    'APP_DEBUG' => false,            //关闭调试模式
);
?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，在此方法中没有任何其他操作，直接指定模板页 index.html，在模板页中创建表单，添加文件域，将图片提交到 upload()方法中进行处理。index()方法的代码如下：

```
public function index(){
    $this->display('index');
}
```

在 Index 控制器的 Index 模块中，定义 upload()方法，实现文件上传的功能。首先，载入上传文件类，实例化上传文件类；然后，对上传文件的大小、类型、存储路径进行设置；接着，调用上传文件类中的 upload()方法执行文件的上传操作，如果上传成功，则将上传文件的信息存储到指定的数据表中，并且在 success.html 模板文件中输出上传图片；否则输出上传失败的错误信息，并且跳转到 index.html 页。其完整代码如下：

```
public function upload(){
        import("ORG.Net.UploadFile");                          //载入上传文件类
        $upload = new UploadFile();                            //实例化上传类
        $upload->maxSize   = 3145728 ;                         //设置附件上传大小
        $upload->allowExts   = array('jpg', 'gif', 'png', 'jpeg');   //设置附件上传类型
        $upload->savePath =   './Public/Uploads/';             //设置附件上传目录
        if($upload->upload()) {                                //上传成功
            $info =   $upload->getUploadFileInfo();
            $upfile = M("files");                              //实例化 User 对象
            $upfile->create();                                 //创建数据对象
            $upfile->tb_name = $info[0]["savename"];           //保存上传图片名称
            $upfile->tb_path = $info[0]["savepath"];           //保存上传图片路径
            $upfile->tb_date = date("Y-m-d H:i:s");            //保存上传图片时间
            $upfile->add();                                    //写入用户数据到数据库
            $db = M('files');                   //实例化模型类，参数数据表名称，不包含前缀
            $select = $db->order('id desc')->select();         //执行查询语句
            $this->assign('select',$select);                   //模板变量赋值
            $this->success("文件上传成功！");                  //上传成功
        }else{
            $info=$upload->getErrorMsg();                      //上传失败返回错误信息
            $this->redirect('index','', 5,$info);              //页面重定向
        }
  }
```



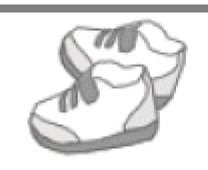

```
        'DB_TYPE'=> 'pdo',
        // 注意 DSN 的配置针对不同的数据库有所区别
        'DB_DSN'=> 'mysql:host=localhost;dbname=db_database15',
        'DB_USER'=>'root',
        'DB_PWD'=>'111',
        'DB_PREFIX'=>'think_',
        // 其他项目配置参数……
        'APP_DEBUG' => false,           //关闭调试模式
    );
    ?>
```

（5）定位到\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，在此方法中没有任何其他操作，直接指定模板页 index.html，在模板页中创建表单，添加文件域，将图片提交到 upload()方法中进行处理。index()方法的代码如下：

```
    public function index(){
      $this->display('index');
    }
```

在 Index 控制器的 Index 模块中，定义 upload()方法，实现文件上传的功能。首先，载入上传文件类，实例化上传文件类；然后，对上传文件的大小、类型、存储路径进行设置；接着，调用上传文件类中的 upload()方法执行文件的上传操作，如果上传成功，则将上传文件的信息存储到指定的数据表中，并且在 success.html 模板文件中输出上传图片；否则输出上传失败的错误信息，并且跳转到 index.html 页。其完整代码如下：

```
    public function upload(){
        import("ORG.Net.UploadFile");                              //载入上传文件类
        $upload = new UploadFile();                                //实例化上传类
        $upload->maxSize   = 3145728 ;                             //设置附件上传大小
        $upload->allowExts   = array('jpg', 'gif', 'png', 'jpeg'); //设置附件上传类型
        $upload->savePath =   './Public/Uploads/';                 //设置附件上传目录
        if($upload->upload()) {                                    //上传成功
            $info =   $upload->getUploadFileInfo();
            $upfile = M("files");                                  //实例化 User 对象
            $upfile->create();                                     //创建数据对象
            $upfile->tb_name = $info[0]["savename"];               //保存上传图片名称
            $upfile->tb_path = $info[0]["savepath"];               //保存上传图片路径
            $upfile->tb_date = date("Y-m-d H:i:s");                //保存上传图片时间
            $upfile->add();                                        //写入用户数据到数据库
            $db = M('files');                    //实例化模型类，参数数据表名称，不包含前缀
            $select = $db->order('id desc')->select();             //执行查询语句
            $this->assign('select',$select);                       //模板变量赋值
            $this->success("文件上传成功！");                       //上传成功
        }else{
            $info=$upload->getErrorMsg();                          //上传失败返回错误信息
            $this->redirect('index','', 5,$info);                  //页面重定向
        }
    }
```





（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文；创建表单，定义文件域，将上传图片提交到 upload()方法中进行处理。其关键代码如下：

```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<form action="__URL__/upload" method="post" enctype="multipart/form-data" name="form1" id=
"form1">
  <tr>
    <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">
      选择文件：
        <input name="tb_path" type="file" id="tb_path" size="15" />
       <input type="submit" name="button" id="button" value="提交" />
    </td>
  </tr>
</form>
```

（7）定位到 App\Tpl\default 目录下，创建 Public 模块文件夹。编辑 success 操作的模板文件 success.html。通过 foreach 标签循环输出模板变量传递的数据，包括上传图片的名称，以及上传图片在服务器中的存储位置；通过 img 标签根据模板变量传递的名称和路径输出上传的图片。其代码如下：

```
<foreach name="select" item="select" >
  <tr class="title">
    <td bgcolor="#FFFFFF">上传图片：{$select.tb_name}</td>
  </tr>
  <tr class="title">
    <td bgcolor="#FFFFFF" width="44"><img src="__ROOT__{$select.tb_path}{$select.tb_name}"
/></td>
  </tr>
</foreach>
```

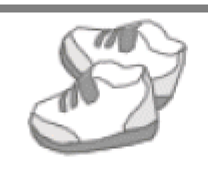
技术要点

本实例主要应用文件上传扩展类（Lib\ORG\Net\UploadFile.class.php）中的方法，实现文件上传的功能。

在 UploadFile 类中，可以对上传文件的属性（参数）进行控制，其可控属性如表 15.3 所示。

表 15.3　在 UploadFile 类中上传文件的可控属性

属　　性	描　　述
maxSize	控制上传文件的大小（以字节为单位），默认为-1（不限制大小）
savePath	控制上传文件在服务器中的存储位置，如果设置为空，则使用 UPLOAD_PATH 常量定义的路径
saveRule	上传文件的保存规则，必须是一个无需任何参数的函数名，例如，time、uniqid com_create_guid 等，但必须保证生成的文件名是唯一的，默认是 uniqid
hashType	传文件的哈希验证方法，默认是 md5_file
autoCheck	是否自动检测附件，默认为自动检测





（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文；创建表单，定义文件域，将上传图片提交到 upload() 方法中进行处理。其关键代码如下：

```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<form action="__URL__/upload" method="post" enctype="multipart/form-data" name="form1" id=
"form1">
  <tr>
    <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">
       选择文件：
         <input name="tb_path" type="file" id="tb_path" size="15" />
        <input type="submit" name="button" id="button" value="提交" />
    </td>
  </tr>
 </form>
```

（7）定位到 App\Tpl\default 目录下，创建 Public 模块文件夹。编辑 success 操作的模板文件 success.html。通过 foreach 标签循环输出模板变量传递的数据，包括上传图片的名称，以及上传图片在服务器中的存储位置；通过 img 标签根据模板变量传递的名称和路径输出上传的图片。其代码如下：

```
<foreach name="select" item="select" >
  <tr class="title">
    <td bgcolor="#FFFFFF">上传图片：{$select.tb_name}</td>
  </tr>
  <tr class="title">
    <td bgcolor="#FFFFFF" width="44"><img src="__ROOT__{$select.tb_path}{$select.tb_name}"
/></td>
  </tr>
</foreach>
```

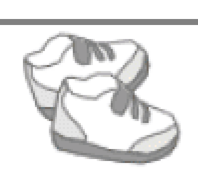
技术要点

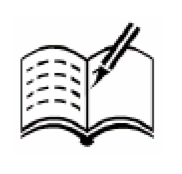
本实例主要应用文件上传扩展类（Lib\ORG\Net\UploadFile.class.php）中的方法，实现文件上传的功能。

在 UploadFile 类中，可以对上传文件的属性（参数）进行控制，其可控属性如表 15.3 所示。

表 15.3　在 UploadFile 类中上传文件的可控属性

属　性	描　述
maxSize	控制上传文件的大小（以字节为单位），默认为-1（不限制大小）
savePath	控制上传文件在服务器中的存储位置，如果设置为空，则使用 UPLOAD_PATH 常量定义的路径
saveRule	上传文件的保存规则，必须是一个无需任何参数的函数名，例如，time、uniqid com_create_guid 等，但必须保证生成的文件名是唯一的，默认是 uniqid
hashType	传文件的哈希验证方法，默认是 md5_file
autoCheck	是否自动检测附件，默认为自动检测





Note

续表

属　性	描　述
uploadReplace	存在同名文件是否是覆盖
allowExts	允许上传的文件后缀（留空为不限制），使用数组设置，默认为空数组
allowTypes	允许上传的文件类型（留空为不限制），使用数组设置，默认为空数组
thumb	是否需要对图片文件进行缩略图处理，默认为 false
thumbMaxWidth	缩略图的最大宽度，多个使用逗号分隔
thumbMaxHeight	缩略图的最大高度，多个使用逗号分隔
thumbPrefix	缩略图的文件前缀，默认为 thumb_
thumbSuffix	缩略图的文件后缀，默认为空
thumbPath	缩略图的保存路径，留空的话取文件上传目录本身
thumbFile	指定缩略图的文件名
thumbRemoveOrigin	指定缩略图的文件名
thumbRemoveOrigin	生成缩略图后是否删除原图
autoSub	是否使用子目录保存上传文件
subType	子目录创建方式，默认为 hash，可以设置为 hash 或者 date
dateFormat	子目录方式为 date 的时候指定日期格式
hashLevel	子目录保存的层次，默认为一层

上传文件的属性设置完成后，应用 upload()方法完成文件上传操作。

上传成功后，应用 getUploadFileInfo()方法获取附件信息。其返回值是一个数组，数组中元素的含义如下：

☑ key：上传的表单名称。

☑ savepath：上传文件的保存路径。

☑ name：上传文件的原始名称。

☑ savename：上传文件的保存名称。

☑ size：上传文件的大小。

☑ type：上传文件的 MIME 类型。

☑ extension：上传文件的后缀类型。

☑ hash：上传文件的哈希验证字符串。

如果上传失败，则应用 getErrorMsg()方法获取错误提示信息。

指点迷津：

在创建文件上传的表单时，在 Form 标签中必须设置 enctype="multipart/form-data"，否则文件不能上传。



PHP 的字符编码

本章读者可以学到如下实例：

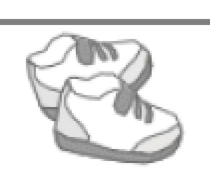
- 实例 259　设计 GB2312 编码格式的网页
- 实例 260　设计 GBK 编码格式的网页
- 实例 261　设计 UTF-8 编码格式的网页
- 实例 262　通过 iconv()函数实现编码格式的转换
- 实例 263　避免截取超长文本时出现乱码
- 实例 264　对输出的数据进行编码格式转换
- 实例 265　论坛中控制帖子标题输出的长度



PHP 的字符编码

本章读者可以学到如下实例：

- ⏭ 实例 259　设计 GB2312 编码格式的网页
- ⏭ 实例 260　设计 GBK 编码格式的网页
- ⏭ 实例 261　设计 UTF-8 编码格式的网页
- ⏭ 实例 262　通过 iconv()函数实现编码格式的转换
- ⏭ 实例 263　避免截取超长文本时出现乱码
- ⏭ 实例 264　对输出的数据进行编码格式转换
- ⏭ 实例 265　论坛中控制帖子标题输出的长度





实例 259　设计 GB2312 编码格式的网页

（实例位置：配套资源\SL\16\259）

实例说明

本实例主要讲解如何设置网页的页面编码为 GB2312 编码格式，运行本实例，如图 16.1 所示，在页面中打印出明日企业网的页面编码的提示信息，从运行结果可以看出该页面所使用的编码为 GB2312 编码格式。



图 16.1　打印明日企业网的页面编码

实现过程

具体步骤如下：

（1）创建 index.php 页面，并设置页面文件的编码为 GB2312，其具体设置方法可参见下面“技术要点”中的相关讲解，这里不再赘述。

（2）在 index.php 文件中通过 HTML 语言的 meta 标记设置网页编码为 GB2312 格式，具体实现代码如下：

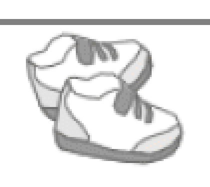
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>设计 GB2312 编码格式的网页</title>
        <meta http-equiv="content-type" content="text/html; charset=gb2312" />
        <link href="css/style.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <br/>
        <div style="width: 487px; text-align:center; height:221px; line-height:221px; font-size:16px; color:blue; background:url(img/bg.jpg)">
            这是一个 GB2312 编码格式的页面
        </div>
    </body>
</html>
```

技术要点

设计 GB2312 编码格式的网页需要对页面进行两点设置，首先使用编辑器设置页面文件的编码为 GB2312 格式，然后通过 HTML 语言的 meta 标记或 PHP 的 header()函数设置页面编码为 GB2312。

设置页面文件编码可以使用 UltraEdit、Edit Plus 和 Zend Studio 等编辑器。下面将以 Zend Studio 为例讲解如何设置页面编码。

打开 Zend Studio 后，选择要设置编码的文件，然后单击鼠标右键，在弹出的快捷菜单中选择“属性”命令（见图 16.2），将弹出如图 16.3 所示的页面属性对话框。





实例 259　设计 GB2312 编码格式的网页

（实例位置：配套资源\SL\16\259）

实例说明

本实例主要讲解如何设置网页的页面编码为 GB2312 编码格式，运行本实例，如图 16.1 所示，在页面中打印出明日企业网的页面编码的提示信息，从运行结果可以看出该页面所使用的编码为 GB2312 编码格式。



图 16.1　打印明日企业网的页面编码

实现过程

具体步骤如下：

（1）创建 index.php 页面，并设置页面文件的编码为 GB2312，其具体设置方法可参见下面“技术要点”中的相关讲解，这里不再赘述。

（2）在 index.php 文件中通过 HTML 语言的 meta 标记设置网页编码为 GB2312 格式，具体实现代码如下：

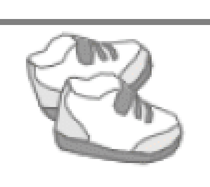
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>设计 GB2312 编码格式的网页</title>
        <meta http-equiv="content-type" content="text/html; charset=gb2312" />
        <link href="css/style.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <br/>
        <div style="width: 487px; text-align:center; height:221px; line-height:221px; font-size:16px; color:blue; background:url(img/bg.jpg)">
            这是一个 GB2312 编码格式的页面
        </div>
    </body>
</html>
```

技术要点

设计 GB2312 编码格式的网页需要对页面进行两点设置，首先使用编辑器设置页面文件的编码为 GB2312 格式，然后通过 HTML 语言的 meta 标记或 PHP 的 header()函数设置页面编码为 GB2312。

设置页面文件编码可以使用 UltraEdit、Edit Plus 和 Zend Studio 等编辑器。下面将以 Zend Studio 为例讲解如何设置页面编码。

打开 Zend Studio 后，选择要设置编码的文件，然后单击鼠标右键，在弹出的快捷菜单中选择“属性”命令（见图 16.2），将弹出如图 16.3 所示的页面属性对话框。



实例 259　设计 GB2312 编码格式的网页

（实例位置：配套资源\SL\16\259）



Note

实例说明

本实例主要讲解如何设置网页的页面编码为 GB2312 编码格式，运行本实例，如图 16.1 所示，在页面中打印出明日企业网的页面编码的提示信息，从运行结果可以看出该页面所使用的编码为 GB2312 编码格式。



图 16.1　打印明日企业网的页面编码

实现过程

具体步骤如下：

（1）创建 index.php 页面，并设置页面文件的编码为 GB2312，其具体设置方法可参见下面“技术要点”中的相关讲解，这里不再赘述。

（2）在 index.php 文件中通过 HTML 语言的 meta 标记设置网页编码为 GB2312 格式，具体实现代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>设计 GB2312 编码格式的网页</title>
        <meta http-equiv="content-type" content="text/html; charset=gb2312" />
        <link href="css/style.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <br/>
        <div style="width: 487px; text-align:center; height:221px; line-height:221px; font-size:16px; color:blue; background:url(img/bg.jpg)">
            这是一个 GB2312 编码格式的页面
        </div>
    </body>
</html>
```

技术要点

设计 GB2312 编码格式的网页需要对页面进行两点设置，首先使用编辑器设置页面文件的编码为 GB2312 格式，然后通过 HTML 语言的 meta 标记或 PHP 的 header()函数设置页面编码为 GB2312。

设置页面文件编码可以使用 UltraEdit、Edit Plus 和 Zend Studio 等编辑器。下面将以 Zend Studio 为例讲解如何设置页面编码。

打开 Zend Studio 后，选择要设置编码的文件，然后单击鼠标右键，在弹出的快捷菜单中选择“属性”命令（见图 16.2），将弹出如图 16.3 所示的页面属性对话框。

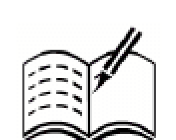






图 16.2　选择“属性”命令

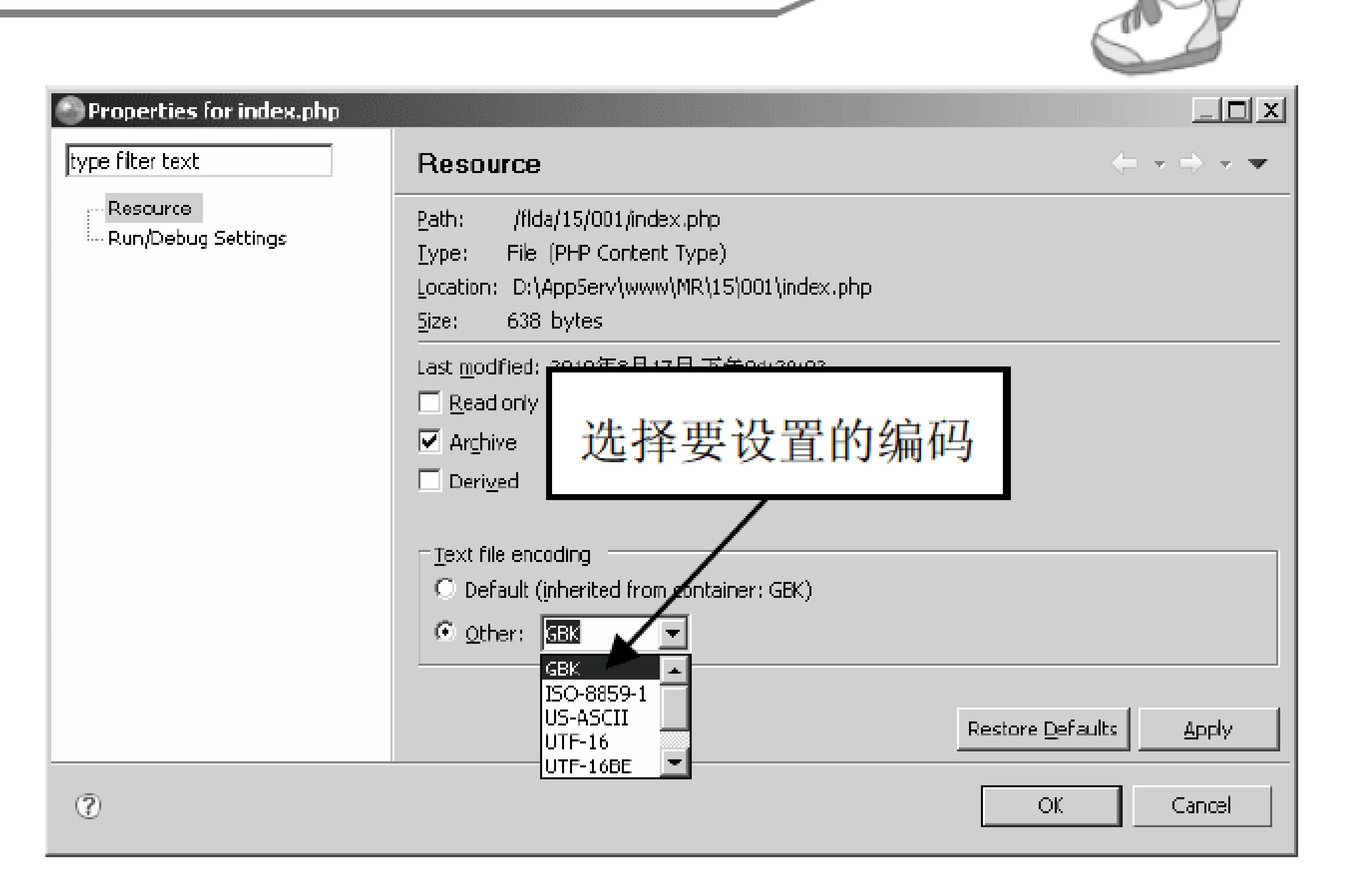


图 16.3　页面属性对话框

在页面属性对话框中选择文件编码模块，在该模块的其他编码的下拉列表框中选择 GBK 编码，然后单击“应用”按钮，最后单击“确定”按钮即可完成对页面文件编码的设置。

指点迷津：

在文件编码下拉列表中并没有 GB2312 编码，而 GBK 编码包含简体中文编码，所以这里选择 GBK 编码。

完成对页面文件的编码设置后，还需要使用 meta 标记设置页面编码，其设置方法如下：

```
<meta http-equiv="content-type" content="text/html; charset=gb2312" />
```

实例 260　设计 GBK 编码格式的网页

（实例位置：配套资源\SL\16\260）

实例说明

本实例主要讲解如何设置网页的页面编码为 GBK 格式。运行本实例，如图 16.4 所示，在页面中打印出咖啡小屋程序所使用的页面编码。

图 16.4　打印咖啡小屋的页面编码

实现过程

具体步骤如下：

（1）创建 index.php 文件，并使用 Zend Studio 编辑器设置页面编码为 GBK 格式。

（2）在 index.php 文件中使用 meta 标记设置网页编码为 GBK 格式，并在页面中打印当前页面所使用的编码，其具体代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

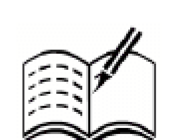





图 16.2　选择“属性”命令

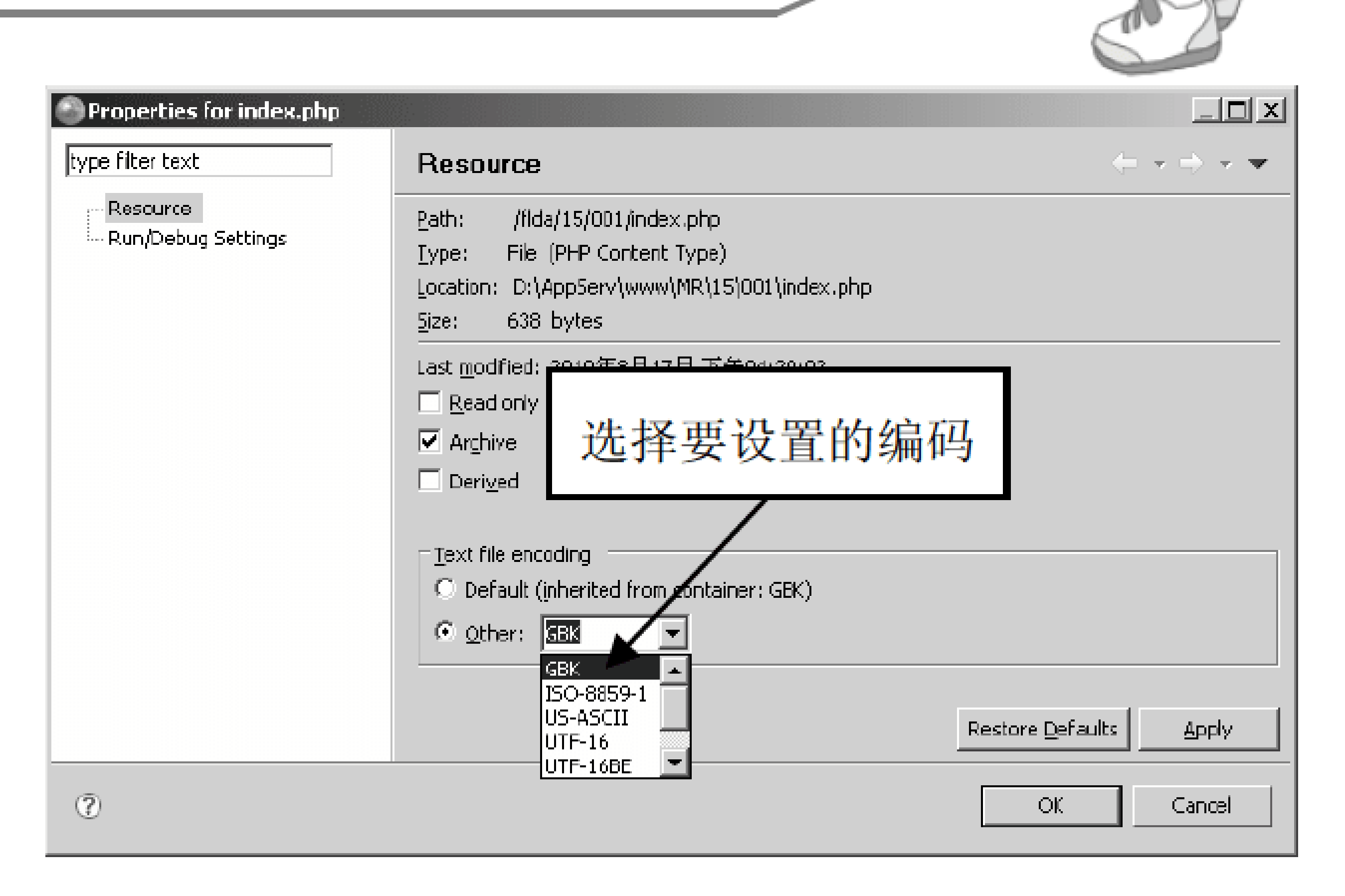


图 16.3　页面属性对话框

在页面属性对话框中选择文件编码模块，在该模块的其他编码的下拉列表框中选择 GBK 编码，然后单击“应用”按钮，最后单击“确定”按钮即可完成对页面文件编码的设置。

指点迷津：

在文件编码下拉列表中并没有 GB2312 编码，而 GBK 编码包含简体中文编码，所以这里选择 GBK 编码。

完成对页面文件的编码设置后，还需要使用 meta 标记设置页面编码，其设置方法如下：

```
<meta http-equiv="content-type" content="text/html; charset=gb2312" />
```

实例 260　设计 GBK 编码格式的网页

（实例位置：配套资源\SL\16\260）

实例说明

本实例主要讲解如何设置网页的页面编码为 GBK 格式。运行本实例，如图 16.4 所示，在页面中打印出咖啡小屋程序所使用的页面编码。

图 16.4　打印咖啡小屋的页面编码

实现过程

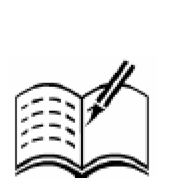
具体步骤如下：

（1）创建 index.php 文件，并使用 Zend Studio 编辑器设置页面编码为 GBK 格式。

（2）在 index.php 文件中使用 meta 标记设置网页编码为 GBK 格式，并在页面中打印当前页面所使用的编码，其具体代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```





```
<head>
    <title>设计 GB2312 编码格式的网页</title>
    <meta http-equiv="content-type" content="text/html; charset=gbk" />
    <link href="css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <br/>
    <div style="width: 498px; text-align:center; height:332px; line-height:221px; font-size:16px; color:#FF3300; background:url(img/bg.jpg)">
        这是一个 GBK 编码格式的页面
    </div>
</body>
</html>
```

技术要点

本实例的关键技术是设置网页的页面编码为 GBK 格式，同样其设置方法也包括两个部分。首先设置网页文件的编码为 GBK 格式，其设置方法与上例相同，这里不再赘述，然后使用 HTML 语言的 meta 标记设置网页的编码为 GBK。其设定代码如下：

```
<meta http-equiv="content-type" content="text/html; charset=gbk" />
```

实例 261　设计 UTF-8 编码格式的网页

（实例位置：配套资源\SL\16\261）

实例说明

本实例主要讲解如何设置网页的页面编码为 UTF-8 格式。运行本实例，如图 16.5 所示，将在页面中打印出企业邮局收发系统所使用的页面编码。

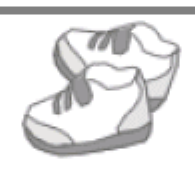


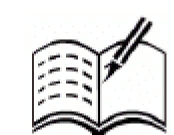
图 16.5　打印企业邮局收发系统的页面编码

实现过程

具体步骤如下：

（1）创建 index.php 文件，并使用 Zend Studio 编辑器设置页面编码为 UTF-8 格式。





（2）在 HTML 语言的 meta 标记中设定当前页面所使用的编码为 UTF-8 格式，其具体代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>设计 UTF-8 编码格式的网页</title>
        <meta http-equiv="content-type" content="text/html; charset=utf-8" />
        <link href="css/style.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <br/>
        <div style="width: 978px; text-align:center; height:591px; line-height:591px; font-size:16px; color:#09600C; background:url(img/bg.jpg)">
            这是一个 UTF-8 编码格式的页面
        </div>
    </body>
</html>
```

技术要点

UTF 是国际字符集转换格式（Unicode/UCS Transformation Format）的缩写，是 FCS 的实际表达式，按其基本长度所用位数分为 UTF-8/16/32 这 3 种形式，其中 UTF-8 保持字符数字一个字节，其他的用不定长编码到最多 6 个字节，支持到 31 位编码。UTF 中最常用的就是 UTF-8 编码格式，UTF-8 使用 8 位字节编码，通常使用 1～3 位字节来表示一个字符，字符的编码长度为变长。

UTF-8 字符集不仅包括欧美字符，也包括中国、日本和韩国的全角字符。因此，在网页中使用 UTF-8 字符集可以使用户在不安装中文编码的情况下浏览到大部分中文字符。

本实例设置页面为 UTF-8 编码，首先需要使用编辑工具设置页面文件编码为 UTF-8 编码，然后使用 HTML 的 meta 标记设置网页编码为 UTF-8 编码，其设置方法如下：

```
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
```

实例 262　通过 iconv()函数实现编码格式的转换

（实例位置：配套资源\SL\16\262　视频位置：配套资源\SP\16\262）

实例说明

在 Dreamweaver 开发工具中，创建一个 UTF-8 编码格式的网页，应用 PHP 中的 setlocale()函数完成区域化设置，输出美式的系统当前时间和中文的系统当前时间。在输出中文的系统当前时间时，因为 strftime()函数的返回值为 GB2312 编码，而当前页面使用的是 UTF-8 编码格式，所以直接输出将会出现乱码，如图 16.6 所示。所以必须应用编码格式转换函数对它的返回值进行编码格式转换才能够正常输出中文格式的时间，这里应用 iconv()函数对



strftime()函数的返回值进行编码格式转换，转换后的运行结果如图 16.7 所示。



图 16.6　输出中文时间乱码



图 16.7　正常输出中文时间

实现过程

本实例的代码如下：

```
<?php
header("Content-Type:text/html;charset=utf-8");
setlocale(LC_ALL, 'en_US');                    //设置为美式英语
echo strftime("%A %e %B %Y", time());          //输出当前时间
echo "<br>";
setlocale(LC_ALL, 'chs');                      //设置为中文
$dates=strftime("%A %e %B %Y", time());        //输出当前时间
echo iconv("gb2312","utf-8",$dates);           //输出经过编码格式转换后的中文时间
?>
```

技术要点

本实例应用 iconv()函数转换编码格式，其语法格式如下：

```
string iconv ( string in_charset, string out_charset, string str )
```

参数 in_charset 为转换输入字符串类型；参数 out_charset 为输出字符串类型；参数 str 为要转换编码风格的字符串集。

实例 263　避免截取超长文本时出现乱码

（实例位置：配套资源\SL\16\263　视频位置：配套资源\SP\16\263）

实例说明

substr()函数是按字节进行截取字符串的，在截取中文字符串时，由于一个汉字是由两个字符组成的，所以如果只截取 1 个字符就会出现乱码。本实例使用自定义函数 msubstr()来解决对中文字符串进行截取时出现乱码的问题，运行结果如图 16.8 所示。

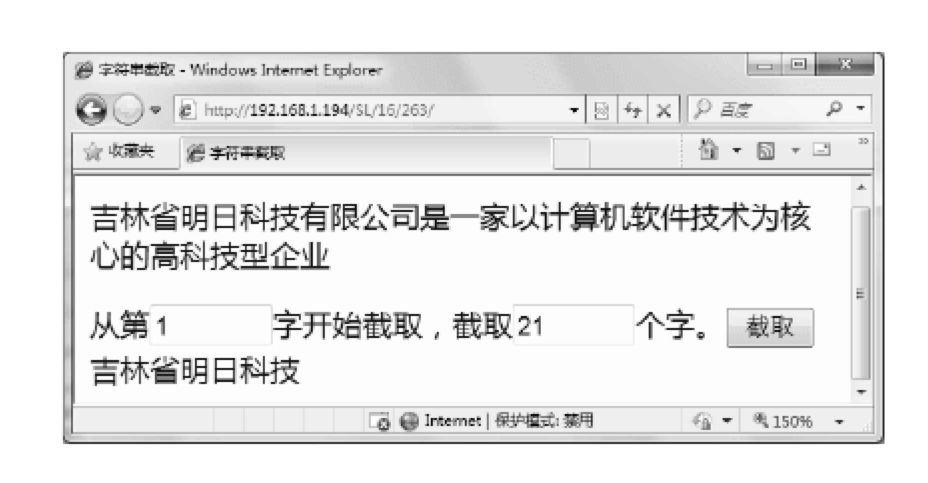
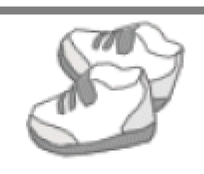
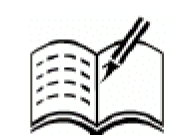


图 16.8　使用 msubstr()函数解决中文字符串截取时的乱码

实现过程

创建 index.php 文件，定义 msubstr()函数，对中文字符串进行截取。创建 form 表单，提交字符串截取的开始位置和长度，然后调用 msubstr()函数对字符串进行截取，并输出截





取结果。代码如下：

```
<?php
function msubstr($str, $start, $len) { //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
  $strlen = $start + $len; //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
  for($i = 0; $i < $strlen; $i ++) {    //通过 for 循环语句，循环读取字符串
        if (ord ( substr ( $str, $i, 1 ) ) > 0xa0) {  //如果字符串中首个字节的ASCII序数值大于0xa0，则表示为汉字
              $tmpstr .= substr ( $str, $i, 2 ); //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
              $i ++; //变量自加 1
        } else {                        //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
              $tmpstr .= substr ( $str, $i, 1 );
        }
  }
  return $tmpstr;                       //输出字符串
}
$string="吉林省明日科技有限公司是一家以计算机软件技术为核心的高科技型企业";
$str=msubstr($string,$_POST[te],$_POST[tx]);
echo $str;
?>
```

技术要点

对中文字符串的截取虽说是通过自定义函数来完成的，但是其根本还是应用 substr 函数，只是在进行字符串截取时，对字符串的类型进行了判断。

对截取字符串中首个字节的 ASCII 序数值进行判断，如果 ASCII 序数值大于 0xa0，则表示为汉字，那么在应用 substr()函数进行截取时，就以两个字节为单位进行截取；如果 ASCII 序数值小于 0xa0，则表示为英文字符串，那么在应用 substr()函数进行截取时，就以 1 个字节为单位进行截取。这样将中文字符串和英文字符串分隔进行截取就避免了出现乱码的问题。

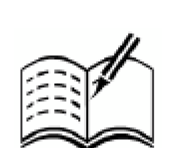
自定义函数 msubstr()的语法如下：

```
function msubstr($str, $start, $len) { //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
  $strlen = $start + $len; //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
  for($i = 0; $i < $strlen; $i ++) {          //通过 for 循环语句，循环读取字符串
        if (ord ( substr ( $str, $i, 1 ) ) > 0xa0)   //如果字符串中首个字节的ASCII序数值大于0xa0，则表示为汉字
              $tmpstr .= substr ( $str, $i, 2 );//每次取出两位字符赋给变量$tmpstr，即等于一个汉字
              $i ++; //变量自加 1
        } else {                  //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
              $tmpstr .= substr ( $str, $i, 1 );
        }
  }
  return $tmpstr;                 //输出字符串
}
```

参数$str 是指定被截取的字符串；参数$strart 是截取的开始位置；参数$len 是截取的



长度。返回值$tmpstr 是截取后的字符串。



Note

实例 264　对输出的数据进行编码格式转换

（实例位置：配套资源\SL\16\264　视频位置：配套资源\SP\16\264）

实例说明

PHP 不仅可以与 MySQL 数据库搭配使用，也可以与其他数据库进行联合应用。在本实例中，通过 PHP 操作 Access 数据库，完成数据库中数据的循环输出，并且通过 iconv()函数对输出数据的编码格式进行转换，运行效果如图 16.9 所示。



书名	出版社
《PHP从入门到精通》	清华大学出版社
《PHP项目开发全程实录》	清华大学出版社
《PHP开发实战宝典》	清华大学出版社

图 16.9　循环输出数据库中的数据

实现过程

具体步骤如下：

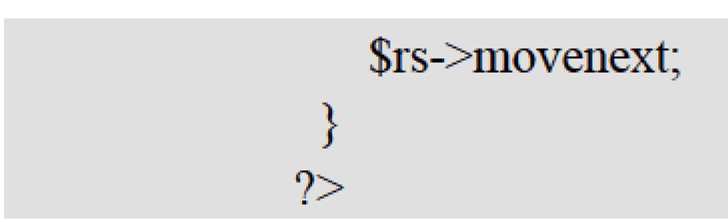
（1）建立数据库连接文件 conn.php 实现与 Access 数据库的连接，代码如下：

```
<?php
$conn = new com("adodb.connection");                    //实例化 com 类
$connstr="driver={microsoft access driver (*.mdb)}; dbq=". realpath("data/db_database16.mdb");
//连接 db_database16 数据库
$conn->open($connstr);                                  //返回连接对象
?>
```

（2）创建 index.php 文件，包含数据库连接文件，调用 ADO 中的方法操作 Access 数据库中的数据，并通过 while 语句循环输出数据库中的数据。其关键代码如下：

```
  <?php
     $sql="select * from tb_book";                      //定义 SQL 语句
     $rs=new com("adodb.recordset");                    //实例化 ADO 中的方法
     $rs->open($sql,$conn,1,3);                         //执行 SQL 语句
     while(!$rs->eof){                                  //循环输出查询结果
 ?>
        <tr>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo iconv('gbk','utf-8',
$rs->fields('bookname')->value);?></div></td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo iconv('gbk','utf-8',
$rs->fields('pub')->value);?></div></td>
        </tr>
    <?php
```





```
            $rs->movenext;
        }
    ?>
```

指点迷津：

本实例中的文件都采用的是 UTF-8 编码格式，但是通过 ADO 中的方法读取到的 Access 数据库中的数据采用的是 GBK 的编码，所以在页面中直接输出时将出现乱码，所以必须通过 iconv()函数对输出的数据进行编码格式的转换才能保证数据的正常输出。

技术要点

在本实例中，PHP 通过预先定义类 com 调用 ADO 方法操纵 Access 数据库。com 类详细说明如下：

```
string com::com( string module_name [, string server_name [, int codepage]])
```

参数说明：

- ☑ module_name：被请求组件的名字或 class-id。
- ☑ server_name：DCOM 服务器的名字。
- ☑ codepage：指定用于将 PHP 字符串转换成 UNICODE 字符串的代码页，反之亦然。该参数的取值有 CP_ACP、CP_MACCP、CP_OEMCP、CP_SYMBOL、CP_THREAD_ACP、CP_UTF7 和 CP_UTF8。

实例 265　论坛中控制帖子标题输出的长度

（实例位置：配套资源\SL\16\265　视频位置：配套资源\SP\16\265）

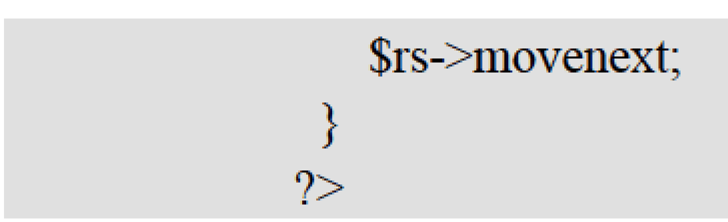
实例说明

在论坛中输出帖子的标题时，为了确保页面的美观及完整性，必须控制超长标题的输出，并以省略号来替换，此时就需要对帖子的标题进行截取，这里推荐读者使用 mb_substr()函数对帖子的标题进行截取，这样可以避免截取中文时出现乱码的问题。在本实例中完成论坛中帖子标题的循环输出，并且对超长的标题进行截取，同时使用省略号进行替换，运行效果如图 16.10 所示。



图 16.10　论坛中控制帖子标题输出的长度





```
            $rs->movenext;
        }
    ?>
```

指点迷津：

本实例中的文件都采用的是 UTF-8 编码格式，但是通过 ADO 中的方法读取到的 Access 数据库中的数据采用的是 GBK 的编码，所以在页面中直接输出时将出现乱码，所以必须通过 iconv()函数对输出的数据进行编码格式的转换才能保证数据的正常输出。

技术要点

在本实例中，PHP 通过预先定义类 com 调用 ADO 方法操纵 Access 数据库。com 类详细说明如下：

```
string com::com( string module_name [, string server_name [, int codepage]])
```

参数说明：

☑ module_name：被请求组件的名字或 class-id。

☑ server_name：DCOM 服务器的名字。

☑ codepage：指定用于将 PHP 字符串转换成 UNICODE 字符串的代码页，反之亦然。该参数的取值有 CP_ACP、CP_MACCP、CP_OEMCP、CP_SYMBOL、CP_THREAD_ACP、CP_UTF7 和 CP_UTF8。

实例 265　论坛中控制帖子标题输出的长度

（实例位置：配套资源\SL\16\265　视频位置：配套资源\SP\16\265）

实例说明

在论坛中输出帖子的标题时，为了确保页面的美观及完整性，必须控制超长标题的输出，并以省略号来替换，此时就需要对帖子的标题进行截取，这里推荐读者使用 mb_substr()函数对帖子的标题进行截取，这样可以避免截取中文时出现乱码的问题。在本实例中完成论坛中帖子标题的循环输出，并且对超长的标题进行截取，同时使用省略号进行替换，运行效果如图 16.10 所示。



图 16.10　论坛中控制帖子标题输出的长度

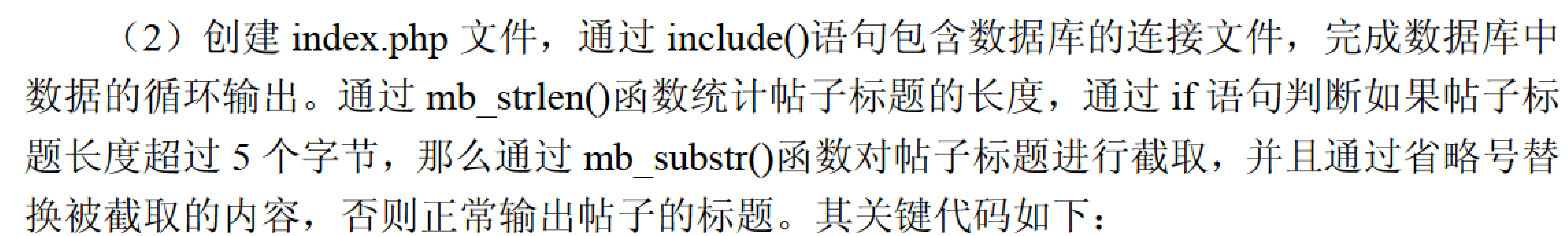


实现过程

具体步骤如下：



（1）编写数据库连接文件 conn.php，同时设置数据库的编码格式为 UTF-8。



（2）创建 index.php 文件，通过 include()语句包含数据库的连接文件，完成数据库中数据的循环输出。通过 mb_strlen()函数统计帖子标题的长度，通过 if 语句判断如果帖子标题长度超过 5 个字节，那么通过 mb_substr()函数对帖子标题进行截取，并且通过省略号替换被截取的内容，否则正常输出帖子的标题。其关键代码如下：

```
<?php
  include_once("conn/conn.php");                                    //载入数据库连接文件
  $query=mysql_query("select * from tb_guestbook",$conn);           //执行查询语句
  while($myrow=mysql_fetch_array($query)){                          //循环输出查询结果
?>
      <tr>
            <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['id'];?></td>
            <td align="center" bgcolor="#FFFFFF">
          <?php
          $str=mb_strlen($myrow['title'],"UTF-8");                  //获取帖子标题的长度
          if($str>5){                                               //判断如果长度超出 5 个字节
                $title=mb_substr($myrow['title'],0,5,"UTF-8");      //对帖子的标题进行截取
                echo $title."...";                                  //输出帖子标题
          }else{
                echo $myrow['title'];                               //输出帖子标题
          }
          ?>
          </td>
            <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['createtime'];?></td>
        </tr>
  <?php }?>
```

技术要点

本实例应用 mb_substr ()函数对字符串进行截取操作，其作用是将指定字符串以特定编码格式从指定位置开始，向后截取指定长度的字节。其语法结构如下：

```
string mb_substr ( string str, int start [, int length [, string encoding]] )
```

该函数与 substr()函数的功能类似。参数 str 为指定被截取的字符串；参数 start 为截取从该字符串的起始位置；参数 length 为指定截取长度；参数 encoding 为指定的编码格式。其中本实例的 encoding 参数为 UTF-8，另外也可以根据实际情况设置该参数为 GB2312 等。

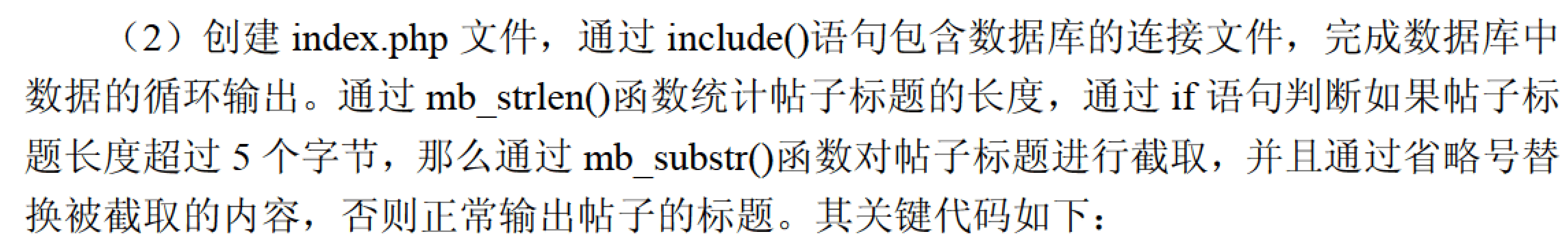


实现过程

具体步骤如下：



（1）编写数据库连接文件 conn.php，同时设置数据库的编码格式为 UTF-8。



（2）创建 index.php 文件，通过 include()语句包含数据库的连接文件，完成数据库中数据的循环输出。通过 mb_strlen()函数统计帖子标题的长度，通过 if 语句判断如果帖子标题长度超过 5 个字节，那么通过 mb_substr()函数对帖子标题进行截取，并且通过省略号替换被截取的内容，否则正常输出帖子的标题。其关键代码如下：

```
<?php
 include_once("conn/conn.php");                              //载入数据库连接文件
 $query=mysql_query("select * from tb_guestbook",$conn);     //执行查询语句
 while($myrow=mysql_fetch_array($query)){                    //循环输出查询结果
?>
    <tr>
         <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['id'];?></td>
         <td align="center" bgcolor="#FFFFFF">
        <?php
        $str=mb_strlen($myrow['title'],"UTF-8");             //获取帖子标题的长度
        if($str>5){                                          //判断如果长度超出 5 个字节
            $title=mb_substr($myrow['title'],0,5,"UTF-8");   //对帖子的标题进行截取
            echo $title."...";                               //输出帖子标题
        }else{
            echo $myrow['title'];                            //输出帖子标题
        }
        ?>
        </td>
         <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['createtime'];?></td>
     </tr>
 <?php }?>
```

技术要点

本实例应用 mb_substr ()函数对字符串进行截取操作，其作用是将指定字符串以特定编码格式从指定位置开始，向后截取指定长度的字节。其语法结构如下：

```
string mb_substr ( string str, int start [, int length [, string encoding]] )
```

该函数与 substr()函数的功能类似。参数 str 为指定被截取的字符串；参数 start 为截取从该字符串的起始位置；参数 length 为指定截取长度；参数 encoding 为指定的编码格式。其中本实例的 encoding 参数为 UTF-8，另外也可以根据实际情况设置该参数为 GB2312 等。